NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

CONSTRUCTION AND ANALYSIS OF MULTI-RATE PARTITIONED RUNGE-KUTTA METHODS

by

Patrick R. Mugg

June 2012

Thesis Advisor:	Francis Giraldo
Second Reader:	Hong Zhou

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE		Form Approved OMB No. 0704–0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE June 2012	3. REPORT TYPE AND DATES COVERED Master's Thesis

4. TITLE AND SUBTITLE Construction and Analysis of Multi-Rate Partitioned Runge-Kutta Methods	5. FUNDING NUMBERS
6. AUTHOR(S): Patrick R. Mugg	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943–5000	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A	10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB Protocol number _____N/A_____.

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited	12b. DISTRIBUTION CODE A

13. ABSTRACT (maximum 200 words)

Adaptive mesh refinement (AMR) of hyperbolic systems allows us to refine the spatial grid of an initial value problem (IVP), in order to obtain better accuracy and improved efficiency of the numerical method being used. However, the solutions obtained are still limited to the local Courant-Friedrichs-Lewy (CFL) time-step restrictions of the smallest element within the spatial domain. Therefore, we look to construct a multi-rate time-integration scheme capable of solving an IVP within each spatial sub-domain that is congruent with that sub-domain's respective time-step size.

The primary objective for this research is to construct a multi-rate method for use with AMR. In this thesis we will focus on constructing a 2^{nd} order, multi-rate partitioned Runge-Kutta ($MPRK_2$) scheme, such that the non-uniform mesh is constructed with the coarse and fine elements at a two-to-one ratio. We will use general 2^{nd} and 4^{th} order finite differences (FD) methods for non-uniform grids to discretize the spatial derivative, and then use this model to compare the $MPRK_2$ time-integrator against three explicit, 2^{nd} order, single-rate time-integrators: Adams-Bashforth 2 (AB_2), Backward Differentiation Formula 2 (BDF_2), and Runge-Kutta 2 (RK_2).

14. SUBJECT TERMS Multi-rate Method, Runge-Kutta, Finite Difference, Time-integration, Partial Differential Equation	15. NUMBER OF PAGES 107
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UU

NSN 7540–01–280–5500 Standard Form 298 (Rev. 2–89)
Prescribed by ANSI Std. 239–18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

CONSTRUCTION AND ANALYSIS OF MULTI-RATE PARTITIONED RUNGE-KUTTA METHODS

Patrick R. Mugg
Captain, United States Army
B.S., United States Military Academy, West Point, 2003

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN APPLIED MATHEMATICS

from the

NAVAL POSTGRADUATE SCHOOL
June 2012

Author: Patrick R. Mugg

Approved by: Professor Francis Giraldo
Thesis Advisor

Associate Professor Hong Zhou
Second Reader

Approved by: Professor Carlos F. Borges
Chair, Department of Applied Mathematics

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Adaptive mesh refinement (AMR) of hyperbolic systems allows us to refine the spatial grid of an initial value problem (IVP), in order to obtain better accuracy and improved efficiency of the numerical method being used. However, the solutions obtained are still limited to the local Courant-Friedrichs-Lewy (CFL) time-step restrictions of the smallest element within the spatial domain. Therefore, we look to construct a multi-rate time-integration scheme capable of solving an IVP within each spatial sub-domain that is congruent with that sub-domain's respective time-step size.

The primary objective for this research is to construct a multi-rate method for use with AMR. In this thesis we will focus on constructing a 2^{nd} order, multi-rate partitioned Runge-Kutta ($MPRK_2$) scheme, such that the non-uniform mesh is constructed with the coarse and fine elements at a two-to-one ratio. We will use general 2^{nd} and 4^{th} order finite differences (FD) methods for non-uniform grids to discretize the spatial derivative, and then use this model to compare the $MPRK_2$ time-integrator against three explicit, 2^{nd} order, single-rate time-integrators: Adams-Bashforth 2 (AB_2), Backward Differentiation Formula 2 (BDF_2), and Runge-Kutta 2 (RK_2).

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF FIGURES

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS, ABBREVIATIONS, AND TERMS

AB_P	Adams-Bashforth method of order p
BDF_P	Backwards Differentiation Formula of order p
Consistency	A FD scheme is said to be consistent if the difference between its approximation to the PDE and the exact solution, i.e., truncation error, goes to zero in the limit, as the grid or mesh is refined.
Explicit	For spatial FD approximations, a method where one unknown derivative value is given by a linear combination of known function values. For FD time-integrators, a method where the unknown (future) function value is given in terms of known function and derivative values.
FD	Finite Difference
Implicit	For spatial FD approximations, a method where linear combinations of unknown derivative values are given in terms of linear combinations of known function values. For FD time-integrators, a method where the unknown (future) function value includes a term involving an unknown (future) derivative value.
Multi-rate	Numerical method that uses multiple time-step values simultaneously, such that each element within the spatial domain uses its congruent time-step size, ensuring local CFL stability
(O)	Big-O notation
ODE	Ordinary Differential Equation
PDE	Partial Differential Equation

RK_M	Runge-Kutta method of order M
Single-rate	Numerical method that uses a single time-step value to advance the solution in time, where the time-step used is limited the local CFL stability restrictions of the smallest element within the grid.
Stability	Numerical methods are considered to be stable if the errors they produce do not grow unbounded as the numerical algorithm is calculated while time increases.
T.E.	Truncation error, which is the difference between the exact solution to the partial derivative and its finite difference representation.

ACKNOWLEDGMENTS

I would like to first thank God for being the center of my life and providing me with the direction and guidance needed to accomplish my work. I would also like to thank my wife, Starrissa, and daughter, Sierra, for their love and support during my graduate studies and for their patience and understanding as I worked to complete this thesis.

I would also like to recognize Dr. Shivasubramanian Gopalakrishnan and Dr. Michal Kopera for their subject matter expertise and help in my research, as well as the rest of the staff and faculty of the Department of Applied Mathematics. Much appreciation is also due to Prof. Hong Zhou, for her advice and constructive reviews to the thesis.

Lastly, I want to acknowledge my thesis advisor, Prof. Frank Giraldo, for his encouragement, expertise and friendship throughout my two years at the Naval Postgraduate School. Without his help, this thesis would not have been possible. Thank you.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

There is an ever increasing need to develop numerical methods capable of solving large-scale, real-world problems. In fact, many of these problems arise in the physical, biological and engineering sciences, such that we must use numerical models, because the problems cannot be solved analytically. In other words, we seek to find numerical methods to approximate the solution to a problem, which is often modeled as a partial differential equation (PDE), or system of PDEs.

For example, the Department of Applied Mathematics of the Naval Postgraduate School in cooperation with the Naval Research Laboratory, both located in Monterey, California are continually refining a modeling environment for solving the three dimensional (3D) compressible Euler and/or Navier-Stokes equations; this is a Non-hydrostatic Unified Model of the Atmosphere, known as NUMA. The NUMA model is constructed using an element-based Galerkin framework, which allows either continuous (CG) or discontinuous (DG) high-order Galerkin methods, and can be used as either a mesoscale or global model, which exhibits excellent scaling properties. Additionally, the model is extremely flexible and allows the user to easily interchange new time-integrators, grid and data structures, or new basis functions, thereby ensuring that the model is algorithmically flexible.

There is a significant interest in using the discontinuous Galerkin method (DG) for solving fluid dynamics problems; studies by [1] and [2], have shown that the DG method is a good choice for the construction of future non-hydrostatic numerical weather prediction models, as it combines high-order accuracy of the solution with geometric flexibility of non-conforming (unstructured) grids. In order to increase the scale resolution capabilities of DG, as well as to take better advantage of available computing power, the use of adaptive mesh refinement (AMR) for a quadrilateral non-conforming grid, is currently being investigated.

In Figures 1 and 2, we see a snapshot of a rising thermal bubble being solved on both an adaptive and uniform mesh, using the quadrilateral based DG method. Figure 1 shows the solution at time $t = 0$ sec., and Figure 2 is at time $t = 1000$ sec.

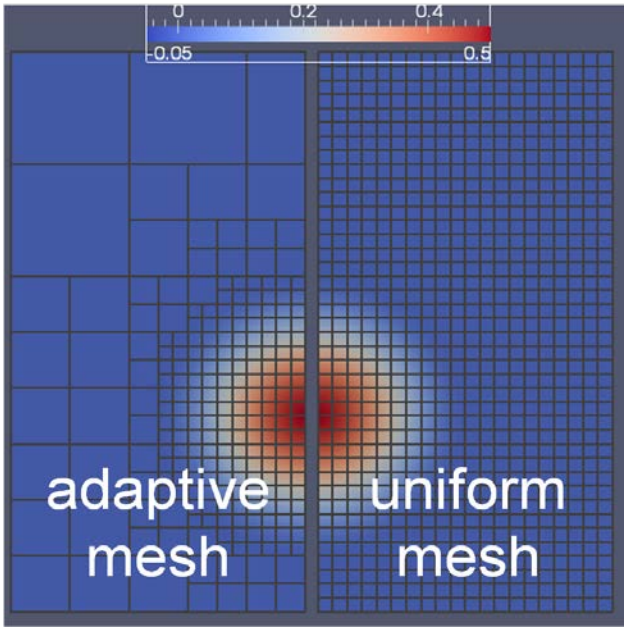

Figure 1. Rising Thermal Bubble (Initial Time = 0 sec.)

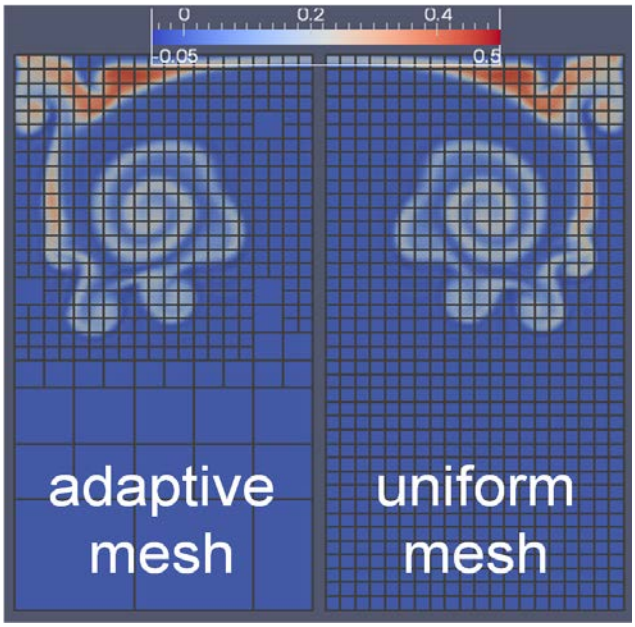

Figure 2. Rising Thermal Bubble (Final Time = 1000 sec.)

To motivate the usefulness of the multi-rate approach let us first describe the AMR strategy. In the figures above, we note that the meshes were refined every second throughout a 1000 second simulation. For example, using a time-step of $\Delta t = 0.005$ would refine the mesh at every 200 time-steps, in order to maintain the desired resolution within the areas of the grid where the temperature variation exists. The refined elements were chosen at the respective time-steps by looking at the value of the temperature within each element and then determining whether that value was below or exceeded a predetermined threshold.

The total runtime of the fully resolved solution (Figure 1, right panel) is approximately 287 seconds compared to the AMR solution with a runtime of 88 seconds. This is a speed-up of nearly 3.26. Even though the AMR solutions require less runtime, the overall numerical method is still restricted to the smallest time-step, which is determined by the smallest grid size within the spatial domain. Therefore, both the uniform and adaptive mesh solutions require the same time-step size. Although we can reduce the computational cost using AMR, we would like to take advantage of the non-conforming grid by simultaneously using larger time-steps within the coarse regions of the spatial domain, and smaller time-steps within the fine regions. This problem of using a time-step size, commensurate with the element size, is the motivation behind my research.

From the literature we find that the current approaches for tackling time-integration include: fully explicit, fully implicit, or a combination of the two, known as implicit/explicit (IMEX) methods. Currently, NUMA uses the IMEX time-integration method, which circumvents the fast and slow propagating waves by splitting up the physical process of the problem being solved, such that the fast moving components are handled implicitly, while the slow moving components are handled explicitly. Although the implicit solutions have no restriction on how large of a time-step one may take, the overall numerical method is still restricted to the time-step size determined by the explicit solution's spatial step-size. Therefore, we look to develop a multi-rate, time-integration method that will allow multiple time-steps to be used simultaneously within the various elements of the spatial domain.

A. OBJECTIVES OF RESEARCH

In this thesis, we will focus on the construction and development of an explicit, second-order, strong stability preserving (SSP) Runge-Kutta (RK), multi-rate method, which we will use to solve the first-order, 1D, advection equation with constant wave speed. This multi-rate method will then be compared to its equivalent single-rate SSP RK_2 method, using both accuracy and efficiency as the two primary metrics.

The emphasis of the thesis is on the time-integration aspects of numerical modeling, and for this reason we restrict our spatial discretization to simple finite difference (FD) methods, albeit of high order, and generalized for non-uniform grid spacing. We will show that the multi-rate approach will retain its formal order of accuracy, while increasing the efficiency of the numerical model. We will show this using complexity analysis (i.e., operations count) and measuring total wall-clock time for the single-rate versus multi-rate approaches.

This thesis is organized into three main chapters, not including the introduction and summary chapters. In Chapter II we will look at how to represent the spatial derivative of our initial value problem (IVP), by constructing a range of finite difference stencils of various orders of accuracy. We will then discuss how to choose which stencil is best applicable by looking at the accuracy, stability and convergence of each spatial stencil, in combination with a forward Euler scheme in time. In Chapter III, we will shift our focus to looking at three different time-integration methods within the multi-step and multi-stage families. Specifically, we will look at the Adams-Bashforth (AB), Backward Differentiation Formula (BDF), and Runge-Kutta (RK) methods (each explicit in time) for representing our temporal derivative. We will solve our IVP using each time-integrator in conjunction with the FD spatial stencils developed in Chapter II. Finally, in Chapter IV we introduce the idea of a non-conforming grid and derive an explicit, second-order, SSP RK multi-rate method. Here, we use the single-rate results from Chapter III in order to compare against our multi-rate method. The thesis will be concluded by a short summary of findings and a discussion on future research.

B. APPLICATION

Throughout this thesis, the governing equation we will use as our test case, is the hyperbolic PDE

$$\frac{\partial^2 u}{\partial t^2} - c^2 \nabla^2 u = 0 \ , \tag{1.1}$$

which is commonly referred to as the *wave equation*, with *c* being the displacement from rest (or wave speed). We chose to use Equation (1.1) because we can easily compute the analytical solution using the method of characteristics, which can only be used for hyperbolic problems possessing the right number of characteristic families. Furthermore, the principal types of problems solved using NUMA are hyperbolic PDEs.

In order to simplify our problem, we will assume that (1.1) is one dimensional, such that

$$\frac{\partial^2 u}{\partial t^2} - c^2 \nabla^2 \mathrm{u} \ = \ \frac{\partial^2 u}{\partial t^2} - c^2 \frac{\partial^2 u}{\partial x^2} \ = \ 0 \ . \tag{1.2}$$

Additionally, we can rewrite equation (1.2) as

$$\left(\frac{\partial}{\partial t} + c \frac{\partial}{\partial x} \right) \left(\frac{\partial}{\partial t} - c \frac{\partial}{\partial x} \right) u = 0 \ , \tag{1.3}$$

since the mixed derivative terms, $\pm c \frac{\partial^2}{\partial t \partial x}$, cancel each other out. If we allow

$$q = \left(\frac{\partial}{\partial t} - c \frac{\partial}{\partial x} \right) u \ \ , \tag{1.4}$$

then after substituting equation (1.4) into equation (1.3) we find the first order, 1D, wave equation, also known as the *advection equation* to be the following:

$$\frac{\partial q}{\partial t} + c \frac{\partial q}{\partial x} = 0 \ . \tag{1.5}$$

Equation (1.5) represents a right moving wave for constant wave speed, such that $c > 0$. Likewise, we could have also chosen to simplify equation (1.3) as a left moving wave;

however, we will now consider Equation (1.5) as the PDE we wish to solve numerically. Therefore, the IVP we wish to solve is equation (1.5), along with the proper initial and boundary conditions, such that the problem being solved is well-posed.

For our purposes, and ease of computation, we impose the following periodic boundary conditions for $x \in [-1,+1]$:

$$q(-1,t) = q(1,t) \tag{1.6}$$

$$\frac{\partial q}{\partial x}(-1,t) = \frac{\partial q}{\partial x}(1,t) \ , \tag{1.7}$$

where these boundary conditions will be helpful as we move from our various single-rate methods to the construction of a multi-rate partitioned Runge-Kutta method, which we will see in Chapter IV. Lastly, we know from the literature, that the analytical solution of our linear and homogeneous IVP, Equation (1.5), can be represented by d' Alembert's solution where we know that the right moving wave can be defined as $q(x,t) = f(x-ct)$.

II. FINITE DIFFERENCES

In this chapter, we will look at the method of finite differences (FD) using our PDE from Chapter I. We will solve the initial-value problem (IVP) using the FD approach, where we will change the domain of the continuous problem into a discretized form. This means that the dependent variable(s) will exist only at the discrete points of the grid generated by our finite difference stencil [3]. Although we know how to solve the original continuous PDE analytically, we will show the benefit in using finite differences to estimate the solution, and how this method can provide an excellent approximation depending on the order of accuracy desired.

The reason for choosing a PDE, in which we know the solution, is to ensure that our numerical method approximates the original PDE well. Once we know that our numerical method is consistent and accurate, we can then apply the method to future PDEs or systems of PDEs where we may not necessarily know how to compute the analytical solution, or finding the exact solution is practically impossible. In these cases, we must instead rely on numerical methods that utilize computer algorithms to approximate the solutions [4].

It is also important to note that in solving an IVP numerically with FD, we must replace all continuous derivatives with finite difference approximations; however, we are not required to use the same stencil or difference quotient for each derivative within the problem. Instead, we are free to choose multiple finite difference schemes for each derivative within the PDE. We will see this later in the chapter; however, let's first take a look at how to build different finite difference schemes for approximating these derivatives.

A. SPATIAL FINITE DIFFERENCE APPROXIMATIONS

Since the linear, first order, 1-D, wave equation has only two first order partial derivatives, one in space, $\partial q/\partial x$, and one in time, $\partial q/\partial t$, we only need to choose two FD stencils. However, we will explore other options for time-integration methods in Chapter

III, where we will see that higher-order FD methods are not as practical in time. For now, we will focus primarily on the spatial derivative $\partial q/\partial x$.

When approximating a derivative using FD, we must make some basic choices that involve how we generate the grid, which will serve as the discretized domain for each solution in space and time where we want to compute the solution of the modified IVP [5]. Clearly, the finer the spatial grid spacing, the more solutions we are required to compute at each time step, and the better our approximation will be to the original continuous problem; however, we must consider the cost-benefit analysis of higher accuracy versus computational cost. It is important to note that a FD stencil with higher accuracy is not only a function of the grid spacing, but also the number of grid points used to approximate the derivative.

Figure 3 shows a general grid using uniform spacing in both space and time. This grid is used to approximate the derivatives for the solution at each grid point (x_i, t^n), using the grid point (x_i, t^n) and its neighboring points. Therefore, if we think of the solution to the PDE as $q(x(t), t)$, then $q_i^n = q(x_i, t^n)$.

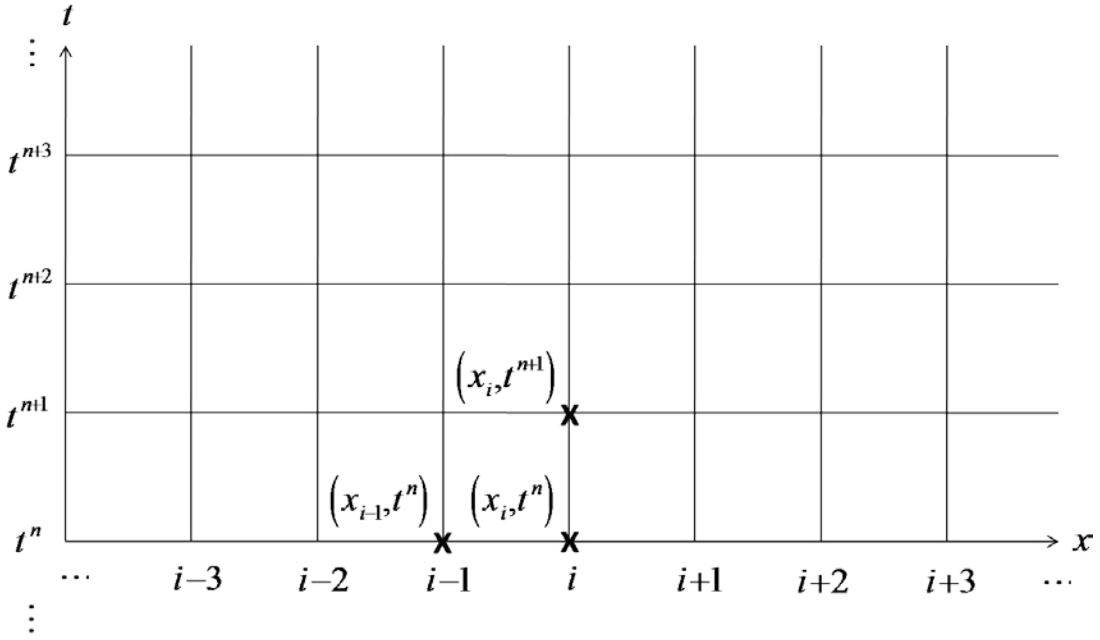

Figure 3. Uniform Grid in Space and Time

Finite difference methods are simple to construct and are beneficial when the geometry of the IVP is regular. Therefore, for our 1-D wave equation we will currently assume we have a uniform grid with periodic boundary conditions for our analysis.

B. FIRST ORDER APPROXIMATION

Using the grid points in Figure 3, we are able to construct a first order backward FD approximation to the continuous spatial derivative at (x_i, t^n), using

$$q_{i-1}^n = q(x_i - \Delta x, t^n)$$

such that

$$\frac{\partial q_i^n}{\partial x} = \frac{q_i^n - q_{i-1}^n}{\Delta x}. \tag{2.1}$$

We begin with the backward-difference formula instead of the forward-difference since it is known in advance that Equation (2.1) is unconditionally stable for the IVP (stability analysis will be a major topic throughout this thesis and will be discussed later in this chapter, and future chapters, for all numerical methods used). Furthermore, this is an obvious choice for the derivative, given that if we take the limit of the right hand side as $\Delta x \to 0$, we have

$$\frac{\partial q_i^n}{\partial x} = \lim_{\Delta x \to 0} \frac{q_i^n - q_{i-1}^n}{\Delta x} + O(\Delta x) = \lim_{\Delta x \to 0} \frac{q(x_i, t^n) - q(x_i - \Delta x, t^n)}{\Delta x} + O(\Delta x)$$

which is the very definition of the first order derivative of q with respect to x at grid point (x_i, t^n). Therefore, we have built a first order approximation to the IVP such that the solution q_i^n is first order accurate $(O(\Delta x))$ in space for a "sufficiently" small but finite Δx [3]. In fact, using Taylor series expansion (TSE) of the backward-difference formula above will provide a more rigorous proof that Equation (2.1) is indeed first order accurate. This can be seen in Appendix A, where we show that the continuous derivative is equivalent to the numerical approximation plus the truncation error (T.E.), such that the T.E. = $O(\Delta x)$.

Note that the forward-difference formula

$$\frac{\partial q_i^n}{\partial x} = \frac{q_{i+1}^n - q_i^n}{\Delta x} + O(\Delta x) \tag{2.2}$$

is also a first order accurate approximation to $\frac{\partial q_i^n}{\partial x}$.

The big *O* notation used above has a very accurate mathematical meaning, such that when we replace T.E. with $O(\Delta x)$, this represents the truncation error as being bounded by a positive real constant *K*, multiplied by the absolute value of $|\Delta x|$, where Δx is reasonably small, such that we can write $|T.E.| \le K|\Delta x|$ as $\Delta x \to 0$. In other words, the truncation error is on the order of Δx raised to the highest power found within all the terms in the truncation error [3]. We can think of (*O*) notation as the order of accuracy for the numerical method, such that if we look at all of the terms in the TSE, and keep only the term with the largest growth rate, then $O(\Delta x)$ means we are keeping all terms of Δx, and throwing away all other terms (Δx^2) and higher. It is important to note that $O(\Delta x)$ does not necessarily inform us on how large the truncation error is, yet it does provide us an indication on how the numerical method performs as Δx gets closer and closer to zero. This is accomplished by refining our grid spacing.

For example, if our numerical method is on the order of $O(\Delta x^2)$, then as we decrease Δx by half, the estimated error of the method will decrease by a factor of four, such that

$$O\left[\left(\frac{\Delta x}{2}\right)^2\right] = O\left(\frac{\Delta x^2}{4}\right).$$

Results for this can be found in Chapter III, where we will look at various numerical time-integration methods for solving our IVP. For more information on (*O*) notation, refer to [6].

C. HIGHER ORDER APPROXIMATIONS

There are many ways we can develop FD approximations for $\partial q/\partial x$. The normal convention for building these representations begins by choosing the specific grid points we want to use to approximate the derivative, and then expanding each point about the grid point (x_i, t^n). The most common first order derivative representations can be found in Table 1, where we only use two to three grid points to represent these derivatives [3].

Derivative		**Finite-Difference Representation**	**Equation**
$\frac{\partial q_i^n}{\partial x}$	=	$\frac{q_i^n - q_{i-1}^n}{\Delta x} + O(\Delta x)$	(2.3)
		$\frac{q_{i+1}^n - q_i^n}{\Delta x} + O(\Delta x)$	(2.4)
		$\frac{q_{i+1}^n - q_{i-1}^n}{2\Delta x} + O(\Delta x^2)$	(2.5)
		$\frac{-3q_i^n + 4q_{i+1}^n - q_{i+2}^n}{2\Delta x} + O(\Delta x^2)$	(2.6)
		$\frac{3q_i^n - 4q_{i-1}^n + q_{i-2}^n}{2\Delta x} + O(\Delta x^2)$	(2.7)

Table 1. Difference Approximations Using Two To Three Points

Equations (2.3) and (2.4) are the backward and forward FD approximations respectively, whereas Equation (2.5) is the centered-difference approximation which can be computed by subtracting the TSE of $q(x_i - \Delta x, t^n)$ from the TSE of $q(x_i + \Delta x, t^n)$ about the point q_i^n. This gives us a second order accurate difference formula for the first derivative using the points q_{i+1}^n and q_{i-1}^n. Appendix B outlines how we can construct a 4th order, centered difference stencil for the first derivative, which will be used in conjunction with the time-integration methods developed in Chapters III and IV to analyze single and multi-rate methods in both uniform and non-uniform grids. This stencil is 4th order, such that

$$\frac{\partial q_i^n}{\partial x} = \frac{q_{i-2}^n - 8q_{i-1}^n + 8q_{i+1}^n - q_{i+2}^n}{12\Delta x} + O(\Delta x^4) \ . \tag{2.8}$$

It is easy to see how we can construct any FD stencil for first, second or even higher order derivatives, as well as mixed partial derivatives, if desired, by using the same approach outlined in Appendix B. However, the PDE we will use only requires first order derivatives; therefore, we will not need to construct any more FD stencils than what has already been shown.

D. DIFFERENCE REPRESENTATION OF A PDE

From Chapter I we saw that the IVP in question is the 1-D, first-order wave equation, with constant wave speed, *c*, such that

$$\frac{\partial q}{\partial t} = -c\frac{\partial q}{\partial x}, \qquad c > 0. \tag{2.9}$$

Using either equations (2.3) and (2.4), or the grid points found in Figure 2, we can rewrite our PDE in a first order accurate discretized form for both space and time. We will use forward Euler to represent the temporal derivative, and Equation (2.3), first order backward-difference formula, for the spatial derivative, such that

$$\frac{\partial q_i^n}{\partial t} = \frac{q_i^{n+1} - q_i^n}{\Delta t} \qquad \text{and} \qquad \frac{\partial q_i^n}{\partial x} = \frac{q_i^n - q_{i-1}^n}{\Delta x}.$$

Substituting these back into Equation (2.9), we then have the following approximation to our PDE:

$$\frac{q_i^{n+1} - q_i^n}{\Delta t} = -c\frac{q_i^n - q_{i-1}^n}{\Delta x}, \tag{2.10}$$

where we can rewrite (2.10) such that

$$q_i^{n+1} = q_i^n - \sigma\left(q_i^n - q_{i-1}^n\right) \qquad , \qquad \text{where } \sigma = c\frac{\Delta t}{\Delta x} \ . \tag{2.11}$$

We will see later on in the chapter, that the value of σ (Courant number) will play an important role in determining where our numerical method is stable, as it measures how fast information flows across the grid points.

Equation (2.11) is the explicit forward Euler (first-order upwind) representation of the PDE (2.9) using a first order FD stencil in space. The method is considered explicit since we are solving the equation for only one unknown value, q_i^{n+1}, which is the solution at the next time step. As previously stated, the spatial and temporal FD schemes do not have to be the same; therefore, if we fix the time stencil to be forward Euler, we have an infinite number of options for representing the spatial derivative. We could choose to use a higher order method in space; however, the overall accuracy of the combined numerical method will only be as good as the weakest link. This means that if we use a first order method in time and a fourth order method in space, the overall method is still only first order accurate. Below are a few more examples for the explicit forward Euler method:

$$\frac{q_i^{n+1} - q_i^n}{\Delta t} = -c \frac{q_{i+1}^n - q_{i-1}^n}{2\Delta x} \tag{2.12}$$

$$\frac{q_i^{n+1} - q_i^n}{\Delta t} = -c \frac{3q_i^n - 4q_{i-1}^n + q_{i-2}^n}{2\Delta x} \tag{2.13}$$

Both methods above are second order accurate in space such that the T.E. for each methods is $O(\Delta t, \Delta x^2)$. This is important, given that we can show that the accuracy in space and time are independent of each other. However, there are many factors we must consider when determining which numerical approximation method to use when solving a particular problem. Of these, accuracy, consistency, stability and efficiency must be analyzed. Although we want a numerical method that is as accurate as possible, consistency and stability are of much more importance, such that if the method is both "consistent and stable, then it will converge to the correct solution" [7]. However, if we have two methods that are both stable, consistent, and have the same order of accuracy, then we might want to look at which method is most efficient (i.e., lower computational cost). It is important to note that regardless of how accurate or efficient the method is, if we do not have both stability and consistency, then our numerical solution will be of no

value [8]. Therefore, we must first ensure the method is stable and consistent, and then we can look for how to increase the accuracy or efficiency of the method, depending on what is of most importance to us.

1. Consistency

We have looked at the accuracy of a single FD stencil; however we have not shown how to determine the accuracy of the entire numerical method. Using Equation (2.10), we can show the accuracy of the method by first applying TSE about the point q_i^n and then substituting into the numerical method.

$$q_i^{n+1} = q(x_i, t+\Delta t) = q_i^n + \frac{\partial q_i^n}{\partial t}\Delta t + \frac{\partial^2 q_i^n}{\partial t^2}\frac{(\Delta t)^2}{2!} + O(\Delta t^3)$$

$$q_{i-1}^n = q(x_i - \Delta x, t^n) = q_i^n - \frac{\partial q_i^n}{\partial x}\Delta x + \frac{\partial^2 q_i^n}{\partial x^2}\frac{(\Delta x)^2}{2!} - O(\Delta x^3)$$

Substituting these expansions into Equation (2.10) gives us the following:

$$\frac{\left(q_i^n + \frac{\partial q_i^n}{\partial t}\Delta t + \frac{\partial^2 q_i^n}{\partial t^2}\frac{(\Delta t)^2}{2!} + O(\Delta t^3)\right) - q_i^n}{\Delta t} = -c\left(\frac{q_i^n - \left(q_i^n - \frac{\partial q_i^n}{\partial x}\Delta x + \frac{\partial^2 q_i^n}{\partial x^2}\frac{(\Delta x)^2}{2!} - O(\Delta x^3)\right)}{\Delta x}\right)$$

where the above equation reduces to:

$$\frac{\partial q_i^n}{\partial t} + \frac{\partial^2 q_i^n}{\partial t^2}\frac{\Delta t}{2!} + O(\Delta t^2) = -c\left(\frac{\partial q_i^n}{\partial x} - \frac{\partial^2 q_i^n}{\partial x^2}\frac{\Delta x}{2!} + O(\Delta x^2)\right)$$

$$\frac{\partial q_i^n}{\partial t} + c\frac{\partial q_i^n}{\partial x} + O(\Delta t, \Delta x) = 0\ . \qquad (2.14)$$

Thus, we have proven that method (2.10) is first order accurate in both space and time.

Now, a numerical method is said to be consistent if the difference between its approximation to the PDE and the exact solution, i.e., truncation error, goes to zero in the limit, as the grid is refined. Therefore, if we allow both the temporal grid spacing

parameter, Δt, and spatial grid spacing parameter, Δx, in Equation (2.14) to both go to zero in the limit, such that $(\Delta t, \Delta x) \to 0$, then we can clearly see we will recover the original continuous PDE, Equation (2.9), thereby showing the numerical method, (2.10), is consistent. This same procedure is valid for any numerical method, where more information on consistency and stability for a generalized discretization can be found in most numerical textbooks.

2. Stability and Convergence

When deciding whether or not a FD stencil is appropriate to use in solving an IVP, we have looked at the properties of accuracy and consistency; however, one of the most critical properties to analyze, is the stability of the method. In fact, not only stability, but also the rate at which the numerical solutions for the method being used converge to the true solution of the PDE.

One way to measure the difference between the true solution, which we will define as ϕ_i^n, and our numerical solution, q_i^n, is to take the norm of the difference between the two solutions at each grid point [9]. Although there are many forms for computing a norm, we will use the Euclidean, or L_2, norm, which is defined as

$$\|q\|_2 \equiv \left(\sum_{i=1}^{N} |q_i|^2 \right)^{1/2}.$$

Therefore, we want to see that in the limit, as $(\Delta t, \Delta x) \to 0$, the value of the norm between the numerical solution and the true solution converges on the order of $O(\Delta t^r, \Delta x^s)$, where *r* and *s* represent the order of accuracy, such that

$$\left\| \phi_i^n - q_i^n \right\|_2 = O(\Delta t^r, \Delta x^s).$$

Note that this numerical method satisfies the *Lax Equivalence Theorem*, "which states that if a FD scheme is linear, stable and accurate of order (r, s), then it is convergent of order (r, s)" [9]. In other words, this theorem establishes the fact that if a unique solution exists for our IVP, and the solution depends continuously on both the

initial data and boundary conditions, then the problem is considered to be well-posed; such that, if we have a consistent and stable method, then the numerical method will indeed converge to the true solution of the original PDE [8].

The above theorem is extremely important and forces us to not simply develop a FD scheme that is accurate or consistent. According to [9], consistency is not enough to assure the convergence of a numerical method. The method must also be *stable*. There are a great number of consistent finite-difference methods that are utterly useless because they are unstable. Therefore, if we return to the definition of stability, we must ensure that the numerical solution, q_i^n, does not grow without bound, such that for every time step $n\Delta t$, less than the final time T, we can find a constant C_T that satisfies $\left\|q^n\right\| \le C_T \left\|q^0\right\|$, for all values of Δt and Δx, that are reasonably small [9].

3. Von Neumann Stability Analysis

We have just seen the importance of stability; however, we would now like to extend this idea to our particular problem at hand, such that using Equation (2.10) or (2.11), we can find for what values of σ our method will be stable. Any value that satisfies the stability conditions will then fall into what we call the stability region.

The most widely known and used procedure for analyzing stability is the Von Neumann Method, such that Von Neumann's stability analysis looks at the discretized solution for a given time step, and represents this solution as a finite Fourier series, where the overall stability of the numerical method can be determined by analyzing each element or Fourier mode of the series [9]. In this manner, we can ensure that the overall stability of our method will indeed be stable if every Fourier mode is stable.

Below is the Fourier series representation for the numerical solution q_j^n,

$$q_j^n = \sum_{l=-N}^{N} \tilde{q}_l^{(n)} e^{ik_l j\Delta x}$$

where $\tilde{q}_l^{(n)}$ represents the amplitudes of the wave, $i=\sqrt{-1}$, k_l are the wave numbers and $k_l\Delta x$ represents the phase angle, sometimes denoted ϕ_l, which determines the low from high frequency waves, such that a phase angle of zero indicates a large amplitude wave and a phase angle of π indicates a small amplitude wave [8].

It would be extremely tedious to evaluate all Fourier modes; therefore, if we instead look at just one, where we ensure that this mode has an amplitude $\tilde{q}^{(n)}$, that does not grow for all time, and is bounded such that $\left|\tilde{q}^{(n)}\right|\le 1$, then we can determine where our region of stability lies. For more detailed information on Von Neumann's Method, look to [3], [8], and [10].

If we now concentrate on the single Fourier mode

$$q_j^n=\tilde{q}^{(n)}e^{ij\phi},$$

then we can substitute this mode into Equation (2.11), with the following statements:

$$q_j^n=G^n e^{ij\phi}, \qquad q_j^{n+1}=G^{n+1}e^{ij\phi}, \qquad q_{j-1}^n=G^n e^{i(j-1)\phi},$$

such that we have

$$G^{n+1}e^{ij\phi}=G^n e^{ij\phi}-\sigma\left(G^n e^{ij\phi}-G^n e^{i(j-1)\phi}\right)$$

where G^n replaces $\tilde{q}^{(n)}$ as the amplitudes. If we now simplify the above expression by dividing through by $G^n e^{ij\phi}$, we find that

$$G=1-\sigma(1-e^{-i\phi}).$$

Using Euler's formula, $e^{\pm i\phi}=\cos\phi\pm i\sin\phi$, offers

$$G=1-\sigma(1-\cos\phi)-i\sigma\sin\phi, \tag{2.15}$$

which we know is the graph of a circle centered at $(1-\sigma,0)$ with radius σ. Note that we want to solve Equation (2.15) for σ values that satisfy $|G|\le 1$ where

$$\sigma \equiv \frac{c\Delta t}{\Delta x} \leq 1. \tag{2.16}$$

From Equation (2.15) we find that the solution is $\sigma \leq 1$ and is referred to as the Courants-Friedrichs-Lewy (CFL) condition, which ensures that our solution for this specific numerical method will be stable within the stability region defined by $\sigma \leq 1$ [10]. This can be seen in Figure 4. This figure depicts the regions of stability for Equation (2.11), such that stability is maintained for values inside of the unit circle (solid blue line), given a particular Courant number. Several values for σ have been provided, which demonstrates that for $0 \leq \sigma \leq 1$, the method is stable.

Another method for analyzing the stability region is to look at the magnitude of the amplification factor $|G|$, such that for $|G| \leq 1$ the method is stable. Figure 5 shows a contour plot, where each contour line represents a value of $|G|$ for $0.5 \leq \sigma \leq 1.15$ and $0 \leq \phi \leq 2\pi$. Here we chose a few values of $\sigma > 1$, in order to verify that they indeed correspond to values of $|G| > 1$, therefore proving that Equation (2.11) is unstable for Courant values greater than one.

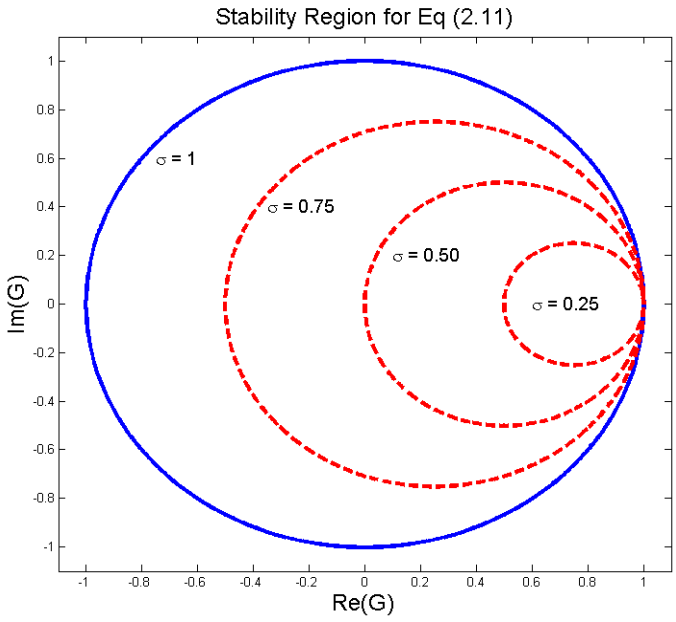

Figure 4. Stability Region for Forward Euler w/ 1st Order Upwind

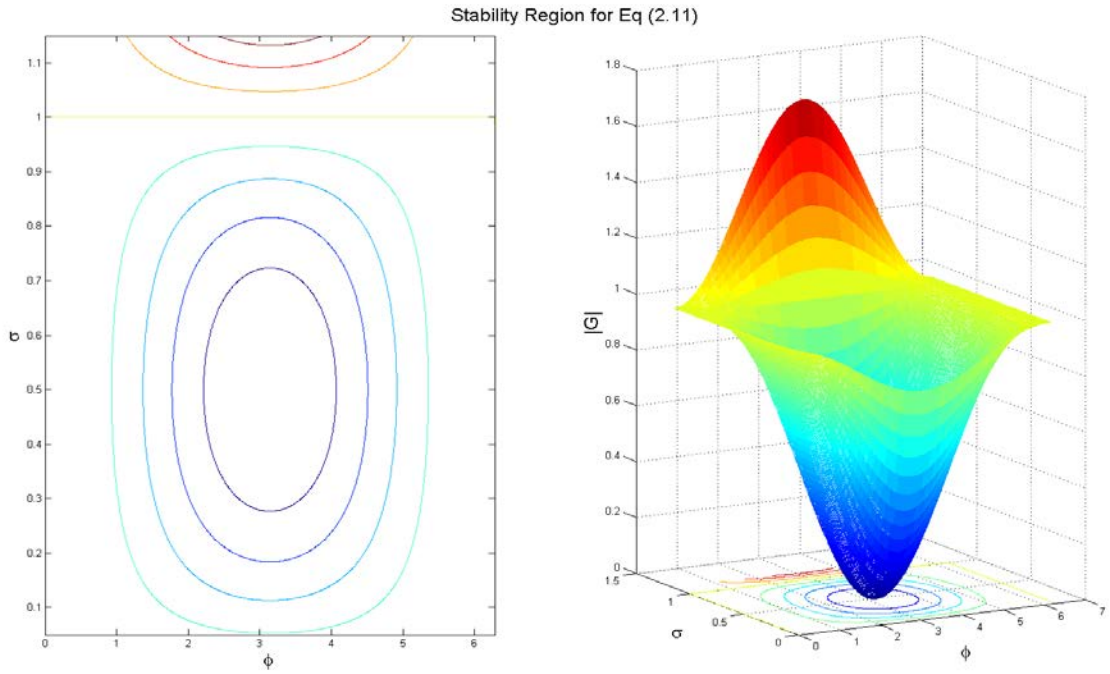

Figure 5. Contour Plot for Forward Euler w/ 1st Order Upwind

THIS PAGE INTENTIONALLY LEFT BLANK

III. TIME-INTEGRATION METHODS

In Chapter II we introduced finite difference (FD) methods and demonstrated how we can use these methods to approximate spatial derivatives. FD are based on Taylor series expansion (TSE), and are also used to derive time-integration stencils, which we will use to solve our initial value problem (IVP). In this chapter, we will turn our focus to two branches of time-integration methods known as multi-step and multi-stage. We will not go into great detail for either of these methods, as they can be found in most numerical analysis texts. However, we will briefly introduce these methods for the purpose of comparing each method with single-rate results using our particular IVP.

A. MULTI-STEP METHODS

Multi-step methods are all methods that require starting values from several previous time steps. These methods can be very useful; however, they require prior information from the system being solved. Therefore, they must first be started by using an initial value, then implementing a one-step method to receive two values, then a two-step method to receive three values, and so on, depending on the order of the method. In fact, Forward Euler (explicit method) and Backward Euler (implicit method) are first-order multi-step methods, although they both contain only one-step.

Other multi-step methods include the Trapezoidal and Leapfrog schemes, which are both implicit, second-order accurate methods. In general, there are two particular families of multi-step methods: the Adams methods and the Backward Differentiation Formulas. We will start by looking at the Adams methods.

1. Adams Methods

Within the Adams family of multi-step methods, the two most commonly used are Adams-Bashforth of order p, (AB_P), and Adams-Moulton of order p, (AM_P), For our

purposes, we want to only focus on explicit time integration methods; therefore, we will only look at AB_P, as these methods are explicit in time, whereas the AM_P are all implicit in time.

The general formula for the Adams-Bashforth method for solving the IVP,

$$q' = F(t,q), \quad q(t_0) = q_0, \quad t \geq t_0,$$

is

$$q^{n+1} = q^n + \Delta t \sum_{j=1}^{m} b_{m-j} F(t^{n+1-j}, q^{n+1-j}) \ , \tag{3.1}$$

such that any AB_P method can be derived by Equation (3.1), which can be constructed using either Taylor series expansion or Lagrange interpolating polynomials. If we turn our focus to AB_2, where $m = 2$, $b_1 = \frac{3}{2}$, and $b_0 = -\frac{1}{2}$ using Equation (3.1), we can see that we achieve the following formula,

$$q^{n+1} = q^n + \frac{\Delta t}{2}\left(3F(t^n, q^n) - F(t^{n-1}, q^{n-1})\right), \tag{3.2}$$

which is a two-step method, and requires the solutions at two previous times, q^n and q^{n-1}, in order to compute the next solution, q^{n+1}, which is at time t^{n+1}. Therefore, we can use any single-step method we like to jumpstart AB_2; however, in order to maintain 2nd order accuracy, the single-step method must also be 2nd order accurate. For a more detailed description of how Equation (3.2) is constructed using TSE, look to Appendix D.

Before using AB_2 with our IVP, it is necessary that we know the stability region of this method; therefore, if we first assume that there is no error within our spatial discretization method, then if we let $F(t^n, q^n) = \lambda q^n$, and $F(t^{n-1}, q^{n-1}) = \lambda q^{n-1}$, then when we perform the Von Neumann stability analysis for AB_2, we have the following:

$$q^{n+1} = q^n + \frac{z}{2}\left(3q^n - q^{n-1}\right), \qquad \text{where } z = \lambda\Delta t$$

$$e^{i(n+1)\theta} - e^{in\theta} = \frac{z}{2}\left(3e^{in\theta} - e^{i(n-1)\theta}\right),$$

where we let $q^n = e^{in\theta}$, because we seek to find the curve for which $\left|q^n\right| = 1$, and defines the boundary of stability. Then, solving for *z*, we find:

$$z = \frac{2(\cos\theta + i\sin\theta) - 2}{3 - \cos\theta + i\sin\theta}.$$

If we now separate the real from the imaginary, we find that:

$$\mathrm{Re}(z) = \frac{-2cos^2\theta + 8cos\theta + 2sin^2\theta - 6}{(3 - cos\theta)^2 + sin^2\theta}$$

,

$$\mathrm{Im}(z) = \frac{-4\sin\theta\cos\theta + 8\sin\theta}{(3 - cos\theta)^2 + sin^2\theta}$$

such that, plotting the $\mathrm{Re}(z)$ versus the $\mathrm{Im}(z)$ provides the region of stability for AB_2, which is shown in Figure 6. From the figure, we see that for any value of $z = \lambda\Delta t$ chosen within this region, the solutions to the IVP will remain stable. However, as stated previously, this is assuming we have zero error within our spatial discretization method. This is important as it allows us to see where each time-integration method is stable, independently of the spatial scheme.

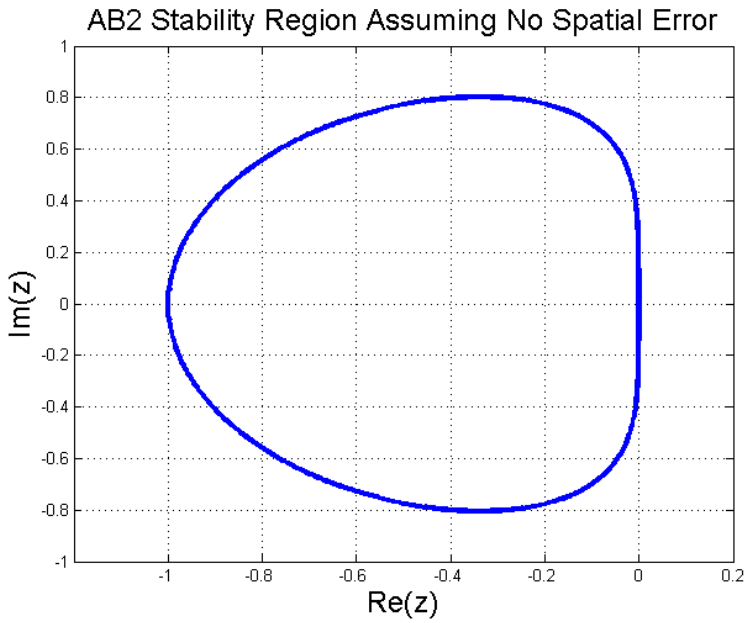

Figure 6. AB_2 Stability Region

We have now seen the general formulas for Forward and Backward Euler, and AB_2. In fact, using the following multi-step formula,

$$q^{n+1} = \sum_{j=0}^{m-1} a_j q^{n-j} + \Delta t \sum_{j=-1}^{m-1} b_j F(t^{n-j}, q^{n-j}) \,, \tag{3.3}$$

if we let $m = 1$, then we have a one-step method, such that Equation (3.3) becomes,

$$q^{n+1} = a_0 q^n + \Delta t \left(b_{-1} F(t^{n+1}, q^{n+1}) + b_0 F(t^n, q^n) \right).$$

Now, if we let $a_0 = b_0 = 1$, and $b_{-1} = 0$, we find that the above equation becomes

$$q^{n+1} = q^n + \Delta t F(t^n, q^n) \,,$$

which is the exact formula for the explicit Euler Method. Therefore, we see that Euler is indeed a multi-step method for *m*-step equal to one, and is equivalently an Adams-Bashforth method of order one (AB_1).

Now, using Equation (3.3) and the coefficients from Table 2, we can formulate a few of the most commonly used multi-step, time-integration methods of order $p = 1, 2, 3$.

Method (*P*)	Coefficients
Euler (1)	$a_0 = b_0 = 1,\ b_{-1} = 0$
Backward Euler * (1)	$a_0 = 1,\ b_{-1} = 1,\ b_0 = 0$
Trapezoidal * (2)	$a_0 = 1,\ b_{-1} = b_0 = \frac{1}{2},\ b_1 = 0$
Leapfrog (2)	$a_0 = 0,\ a_1 = 1,\ b_{-1} = b_1 = 0,\ b_0 = 2$
BDF_2 * (2)	$a_0 = \frac{4}{3},\ a_1 = -\frac{1}{3},\ b_{-1} = \frac{2}{3},\ b_0 = b_1 = 0$
AB_2 (2)	$a_0 = a_1 = b_{-1} = 0,\ b_0 = \frac{3}{2},\ b_1 = -\frac{1}{2}$
AM_2 * (3)	$a_0 = 1,\ ,\ a_1 = 0,\ b_{-1} = \frac{5}{12},\ b_0 = \frac{8}{12},\ b_1 = -\frac{1}{12}$

Table 2. Multi-step Methods of Order p = 1,2,3 *Implicit Method

From Table 2 we also notice that depending on the value of the b_j coefficient, for $j = 1$, then Equation (3.3) will generate either explicit or implicit time-integration methods. In other words, for all values of $b_{-1} \neq 0$, then the multi-step method will be implicit in time.

2. Backward Differentiation Formulas

Another family of multi-step methods is known as the Backward Differentiation Formulas (BDF) of order *p*, such that BDF_2 represents a two-step method, which is second order accurate in time. It is important to note that the number of steps in the method does not necessarily determine the order of accuracy of the method. In general, it is possible to prove that the maximal order of a convergent implicit, *m*-step method, using Equation (3.3), is at most $m+2$ if *m* is even, and $m+1$ if *m* is odd; however, for explicit schemes, an *m*-step method cannot attain an order greater than *m* [11]. This is known as the first Dahlquist barrier. For example, Table 2 lists AM_2 as a two-step method; however, this is an implicit time-integration method with 3rd order accuracy in time.

Although Equation (3.3) is the general equation for all multi-step methods, we saw that there was a more general expression for the family of AB_P methods using Equation (3.1). This is also true of BDF_P, such that we can express every BDF_P method using the following equation:

$$\sum_{j=0}^{m} a_{m-j} q^{n+1-j} = \Delta t F(t^{n+1}, q^{n+1}) \,, \tag{3.4}$$

Previously, we showed that AB_1 was equivalent to Forward Euler (explicit method). In a similar fashion, we can show that Backward Euler (implicit method) is equivalent to BDF_1, for $m = 1,\ a_0 = a_1 = 1$. Yet, of more interest to us, is BDF_2, as this method will provide us with an additional 2nd order time-integration method to use for our IVP. However, given that we want to only focus on explicit time-integrators, we must look at how we can rewrite BDF_2 so that it is instead explicit in time.

First, let us consider BDF_2:

$$q^{n+1} = \frac{4}{3} q^{n} - \frac{1}{3} q^{n-1} + \frac{2\Delta t}{3} F(t^{n+1}, q^{n+1}) \tag{3.5}$$

We can see that the implicit form of BDF_2 requires information from two previous time steps, in addition to the solution at time t^{n+1}. In order to solve for q^{n+1}, we would need to solve a system of equations; therefore, in order to remove the implicitness of the equation, we can instead use an approximation for $F(q^{n+1})$ by the method of extrapolation. Since we can solve the IVP for the solutions at $F(q^n)$ and $F(q^{n-1})$, then if we say that

$$\Delta F^n = F(q^n) - F(q^{n-1}) \ \text{ and } \ \Delta F^{n+1} = F(q^{n+1}) - F(q^n) \,,$$

and then approximate $\Delta F^{n+1} \approx \Delta F^n$, we will find that

$$F(q^{n+1}) - F(q^n) \approx F(q^n) - F(q^{n-1})$$

$$F(q^{n+1}) \approx 2F(q^n) - F(q^{n-1}) \,.$$

However, this approximation uses the assumption that we can use a linear extrapolation. Therefore, if we approximate $F(q^{n+1})$ by $F(q^n)$ and $F(q^{n-1})$ using TSE about $F(q^{n+1})$, we find:

$$(a) \quad F(q^n) = F(q^{n+1}) - \frac{\partial F(q^{n+1})}{\partial t}\Delta t + O(\Delta t^2)$$

$$(b) \quad F(q^{n-1}) = F(q^{n+1}) - 2\frac{\partial F(q^{n+1})}{\partial t}\Delta t + O(\Delta t^2) \ .$$

Now, if we multiply (*a*) by 2 and subtract (*b*), then we get an expression for $F(q^{n+1})$, such that,

$$F(q^{n+1}) = 2F(q^n) - F(q^{n-1}) \ .$$

Although both methods provided the same expression for $F(q^{n+1})$, we have found that the solution using the TSE approach is exact, whereas, we cannot always guarantee the linear extrapolation method will work. If we now substitute this approximation back into Equation (3.5), we have an explicit formula for BDF_2, such that

$$q^{n+1} = \frac{4}{3}q^n - \frac{1}{3}q^{n-1} + \frac{2\Delta t}{3}\left(2F(t^n, q^n) - F(t^{n-1}, q^{n-1})\right) \ . \tag{3.6}$$

Refer to Appendix D for a more thorough derivation of Equation (3.6).

Let us now look at the Von Neumann stability analysis for Equation (3.6), where we again assume there is no spatial error, such that,

$$q^{n+1} = \frac{4}{3}q^n - \frac{1}{3}q^{n-1} + \frac{2z}{3}\left(2q^n - q^{n-1}\right),$$

where we let $q^n = e^{in\theta}$, $q^{n-1} = e^{i(n-1)\theta}$, and $q^{n+1} = e^{i(n+1)\theta}$, such that factoring out the term $e^{in\theta}$ provides the following expression:

$$e^{i\theta} = \frac{4}{3} - \frac{1}{3}e^{-i\theta} + \frac{2z}{3}\left(2 - e^{-i\theta}\right).$$

Now, solving for *z*, we find that

$$z = \frac{\left(\frac{4}{3}\cos\theta - \frac{4}{3}\right) + i\frac{2}{3}\sin\theta}{\left(\frac{4}{3} - \frac{2}{3}\cos\theta\right) + i\frac{2}{3}\sin\theta} \quad .$$

Therefore, separating the real from the imaginary, we finally have the following:

$$\mathrm{Re}(z) = \frac{-2cos^2\theta + 6cos\theta + sin^2\theta - 4}{cos^2\theta - 4\cos\theta + 4} \quad ,$$

$$\mathrm{Im}(z) = \frac{-3\sin\theta\cos\theta + 4\sin\theta}{cos^2\theta - 4\cos\theta + 4}$$

such that, a plot of the stability region for BDF_2 is shown in Figure 7, where we see that for any value of $z = \lambda\Delta t$ chosen within this region, the solutions to the IVP will also remain stable assuming no spatial error.

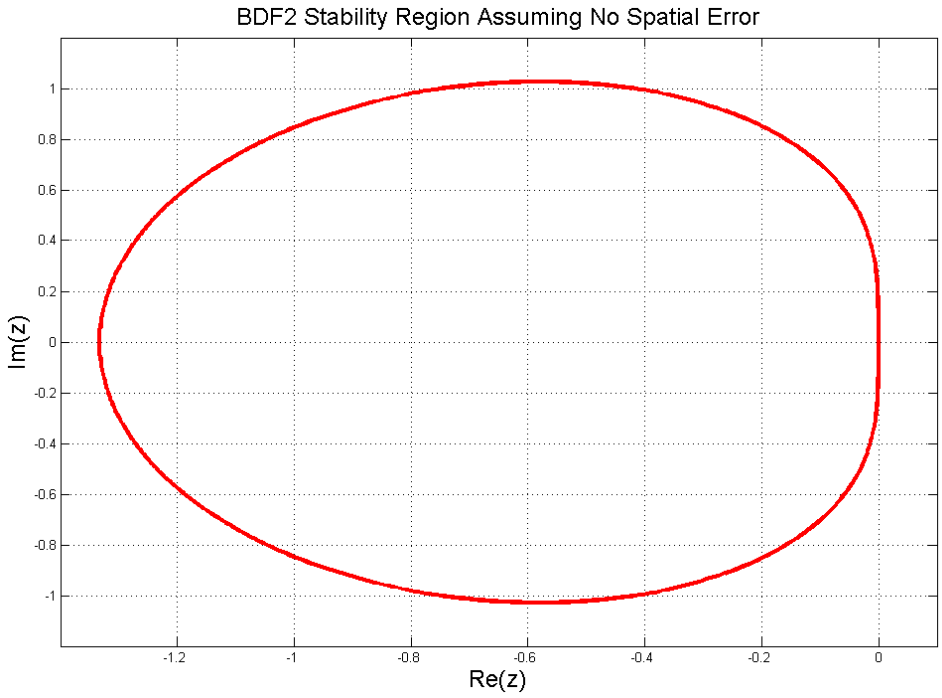

Figure 7. BDF_2 Stability Region

B. MULTI-STAGE METHODS

Like multi-step methods, multi-stage methods have the desirable property that they can achieve very high-order accuracy, while simultaneously reducing the amount of derivatives that need to be computed at each grid point. In order to design a multi-stage method, let us first consider the ordinary differential equation (ODE), $y' = F(t, y)$, such that if we integrate from time t^n to t^{n+1}, then we can show that,

$$y(t^{n+1}) - y(t^n) = \int_{t^n}^{t^{n+1}} y'(t)dt = \int_{t^n}^{t^{n+1}} F(t, y(t))dt .$$

If we now apply the trapezoidal rule, we get

$$y(t^{n+1}) - y(t^n) = \frac{\Delta t}{2}\left(F(t^n, y(t^n)) + F(t^{n+1}, y(t^{n+1}))\right) + O(\Delta t^2),$$

which is the Trapezoidal method in Table 2, such that,

$$y^{n+1} = y^n + \frac{\Delta t}{2}\left(F(t^n, y^n) + F(t^{n+1}, y^{n+1})\right),$$

where the global error is $O(\Delta t^2)$; therefore, this method is a second order, implicit multi-step method. It is important to note, that in general, where we have written Δt, we typically write *h*, such that *h* is any step size, and is not restricted to only time. However, for our purpose in solving the IVP, we will use these methods as time-integrators where we let $h = \Delta t$.

Although we have an expression for an implicit, multi-step method, we instead want an explicit, multi-stage method. To accomplish this, we will now use Forward Euler to solve the ODE for y^{n+1}, such that Euler's method will be called a "predictor," and labeled as $\tilde{y}^{n+1}$. We then substitute $\tilde{y}^{n+1}$ into the Trapezoidal method, known as the "corrector," which gives us the following:

Euler: $\tilde{y}^{n+1} = y^n + \Delta t F(t^n, y^n)$ (predictor)

Trapezoidal: $y^{n+1} = y^n + \frac{\Delta t}{2}\left(F(t^n, y^n) + F(t^{n+1}, \tilde{y}^{n+1})\right)$ (corrector)

The above combination of Euler and Trapezoidal methods is referred to as Heun's method, and is now an explicit, second order, multi-stage method. We may also choose to look at the above method by writing it the following way:

$$\begin{aligned} k_1 &= \Delta t F(t^n, y^n) \\ k_2 &= \Delta t F(t^{n+1}, y^n + k_1) \\ y^{n+1} &= y^n + \frac{1}{2}(k_1 + k_2) \end{aligned} \quad , \tag{3.7}$$

where the number of stages is represented by k_i , (i = 2,…,p) . Here, Heun's method, Equation (3.7), is known as a two-stage method, for $p = 2$.

1. Explicit Runge-Kutta Methods

We have just shown how to construct a very simple, explicit, multi-stage method, where Heun's method is classified within a much larger family of multi-stage methods known as Runge-Kutta of order M (RK_M). Here, Heun's method is most commonly referred to as RK_2.

In general, any explicit p-stage RK_M method can be constructed in the following way:

$$\begin{aligned} k_1 &= \Delta t F(t^n, y^n) \\ k_i &= \Delta t F\left(t^n + d_i \Delta t, y^n + \sum_{j=1}^{i-1} c_{ij} k_j \right) \quad , \quad i = 2, ..., p \, , \\ y^{n+1} &= y^n + \sum_{j=1}^{p} b_j k_j \end{aligned}$$

where any RK_M method is determined by the coefficients p, c, d, and b, which are typically displayed in a table referred to as a Butcher tableau, after J.C. Butcher [10].

d^T	c
	b

p – Number of stages. This is not to be confused with the order of the method; hence, change of subscript to M in RK_M. For example, there are fifth order RK methods with six stages.

c – A p *x* p coefficient matrix.

d – Row vector of size p.

b – Row vector of size p.

Although we will not focus on the derivation of how these parameters are chosen for specific RK_M methods, the reader can find more information in "Numerical Methods for Ordinary Differential Equations," by Butcher [10] or "Numerical Analysis," by Burden and Faires [12].

Below are examples for RK_2, RK_3 and RK_4, with their associated Butcher Tableau:

$$RK_2: \begin{cases} k_1 = \Delta t F(t^n, y^n) \\ k_2 = \Delta t F(t^{n+1}, y^n + k_1) \\ y^{n+1} = y^n + \frac{1}{2}(k_1 + k_2) \end{cases} \qquad \begin{array}{c|cc} 0 & 0 & 0 \\ 1 & 1 & 0 \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array}$$

$$RK_3: \begin{cases} k_1 = \Delta t F(t^n, y^n) \\ k_2 = \Delta t F\left(t^n + \frac{\Delta t}{2}, y^n + \frac{k_1}{2}\right) \\ k_3 = \Delta t F\left(t^n + \Delta t, y^n + k_1 + 2k_2\right) \\ y^{n+1} = y^n + \frac{1}{6}k_1 + \frac{2}{3}k_2 + \frac{1}{6}k_3 \end{cases} \qquad \begin{array}{c|ccc} 0 & & & \\ \frac{1}{2} & \frac{1}{2} & & \\ 1 & 1 & 2 & \\ \hline & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} \end{array}$$

$$RK_4:\begin{cases} k_1 = \Delta t F(t^n, y^n) \\ k_2 = \Delta t F\left(t^n + \dfrac{\Delta t}{2}, y^n + \dfrac{k_1}{2}\right) \\ k_3 = \Delta t F\left(t^n + \dfrac{\Delta t}{2}, y^n + \dfrac{k_2}{2}\right) \\ k_4 = \Delta t F\left(t^n + \Delta t, y^n + k_3\right) \\ y^{n+1} = y^n + \dfrac{1}{6}\left(k_1 + 2k_2 + 2k_3 + k_4\right) \end{cases}$$

0				
$\frac{1}{2}$	$\frac{1}{2}$			
$\frac{1}{2}$	0	$\frac{1}{2}$		
1	0	0	1	
	$\frac{1}{6}$	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{6}$

Recall from Chapter II, if our numerical method is of order $O(h^p)$, then as we reduce the step size by a factor of two, the estimated error of the method is reduced by a factor of 2^p. Therefore, for any *p*-th order method, if we let $p = M$ and $h = \Delta t$, then for RK_4, which has global truncation order of $O(\Delta t^4)$,

$$O\left[\left(\frac{\Delta t}{2}\right)^4\right] = O\left(\frac{\Delta t^4}{16}\right). \tag{3.8}$$

In general, both multi-step and multi-stage methods have their advantages and disadvantages. Higher order multi-step methods require more use of past values, whereas multi-stage methods require more calculations per step. For RK methods, the main computational effort is in the evaluation of the right hand side (RHS) function itself [10]. For example, in the second-order RK methods, the local truncation error is $O(\Delta t^2)$, where the cost is two functional evaluations per step, given there are two stages within the step [10]. Likewise, the fourth-order RK methods require four functional evaluations per step, as there are four stages, and the local truncation error is $O(\Delta t^4)$. However, this pattern does not hold for all RK methods. In general, Table 3 shows "the relationship between the number of evaluations per step and the order of the local truncation error" [10].

Number of Evaluations / Step and Order of the Local Truncation Error

Evaluations per step	2	3	4	$5 \le n \le 7$	$8 \le n \le 9$	$10 \le n$
Best possible local truncation error	$O(\Delta t^2)$	$O(\Delta t^3)$	$O(\Delta t^4)$	$O(\Delta t^{n-1})$	$O(\Delta t^{n-2})$	$O(\Delta t^{n-3})$

Table 3. Evaluations per Step and Truncation Error

Given the result from Equation (3.8), it is easy to see why a higher order RK method might be preferred to a lower-order RK, if we are concerned with accuracy; however, Table 3 shows that in comparing higher-order RK methods ($n \ge 5$) with lower-order RK methods ($n \le 4$), that the lower-order methods may instead be preferable to higher-order. In other words, although the higher order methods allow for a larger time step and improved local truncation error, the overall computational cost increases significantly. Therefore, depending on the problem being solved, we may be satisfied with using a 3rd or 4th order RK method, which requires a slightly smaller time step, compared to say, an 8th order method; however, the overall number of functional evaluations per step is reduced drastically. There is always a tradeoff between choosing a method that has the best possible local truncation error, yet minimizes cost.

For our analysis, we will focus only on explicit, second-order methods; therefore, let us now look at the stability of RK_2, which will be our primary time-integration method for evaluating the IVP in Chapter I. If we rewrite Equation (3.7) in the following way:

$$q^{n+1} = q^n + \frac{\Delta t}{2}\left(F(q^n) + F(q^n + \Delta t F(q^n))\right), \tag{3.9}$$

then, after applying Von Neumann stability analysis to Equation (3.9), we find the subsequent representation for *z*, such that,

$$z^2 + 2z - 2e^{i\theta} + 2 = 0 . \tag{3.10}$$

Here, Equation (3.10) is quadratic; therefore, we must now solve for the roots of the

function, and then plot the real versus imaginary values of z, for $0 \le \theta \le 2\pi$. Figure 8 displays these results, where we now see the stability region for Equation (3.9) assuming no spatial error.

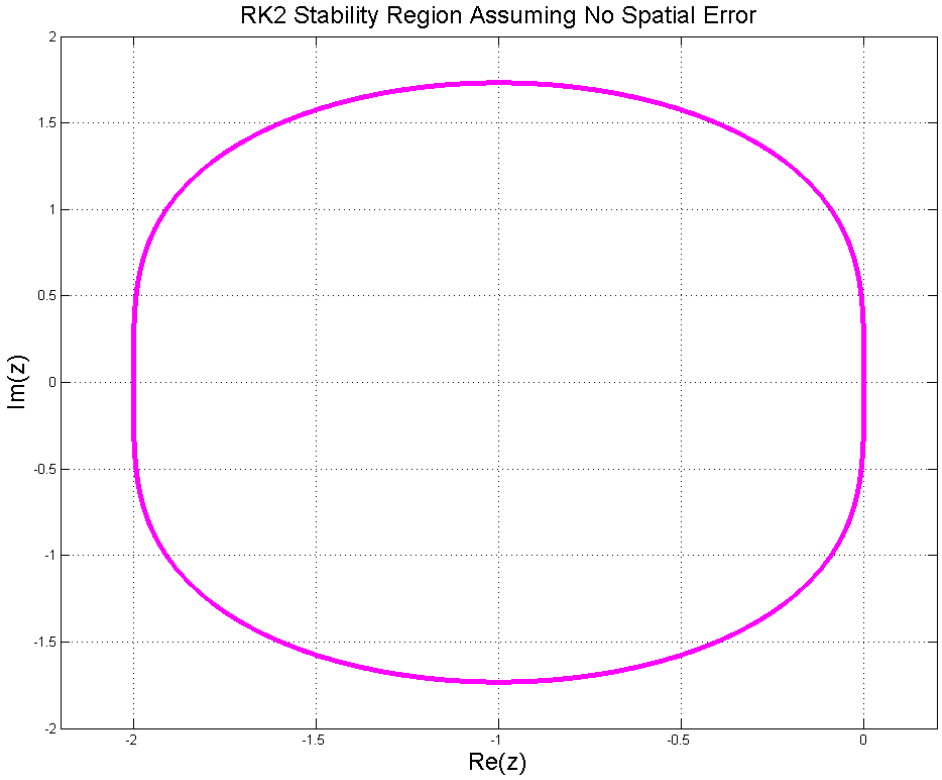

Figure 8. RK_2 Stability Region

Although these plots are useful on their own, it is more beneficial if we overlay each method's stability region, in order that we may compare AB_2, BDF_2, and RK_2 against each other. Figure 9 shows the stability plot for each of these methods, in addition to first-order Euler, where we can easily see that RK_2 has a significantly larger region of stability in comparison to the other three explicit methods. Also, this plot visually supports the fact that RK_2 is stable for much larger time-step values than the other three methods.

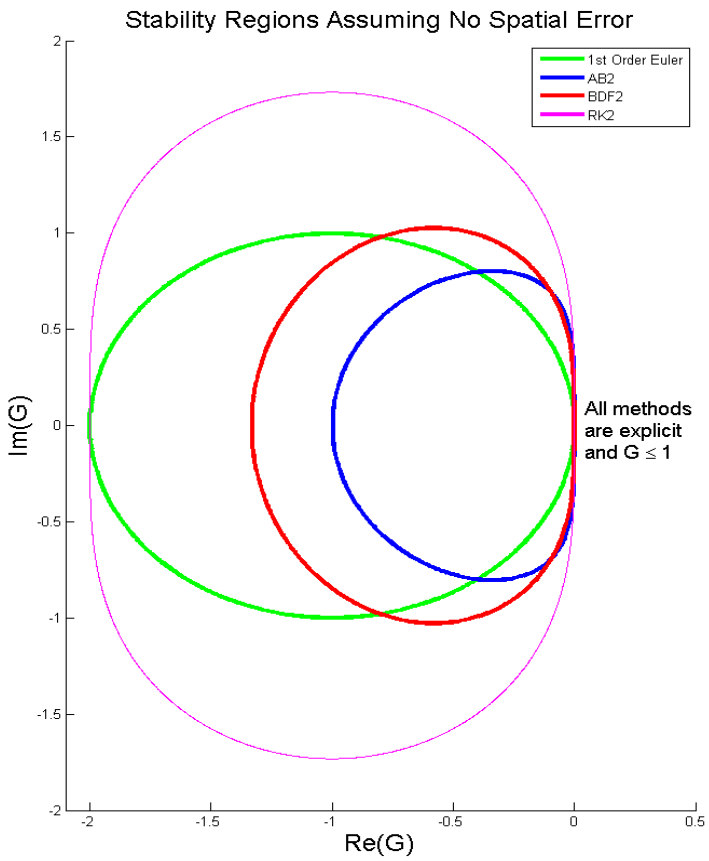

Figure 9. Stability Regions for AB_2, BDF_2, and RK_2

C. SINGLE-RATE RESULTS ON UNIFORM GRID

Let us now look at the numerical results for solving our IVP on a uniform grid, using the multi-step and multi-stage time-integrators mentioned in sections A and B. First, if we choose our initial condition to be a Gaussian function of the form:

$$q(x) = ae^{\frac{-(x-b)^2}{2c^2}}, \tag{3.11}$$

where *a*, *b*, and *c* are real positive constants, such that,

- *a:* height of the curve's peak,
- *b:* center position of curve's peak,
- *c:* width of bell curve,

then we know the general graph of this function is a symmetric bell curve that has tail ends, which fall off to positive and negative infinity. Figure 10 shows three Gaussian functions with varying parameters to demonstrate how each parameter affects the shape of the curve.

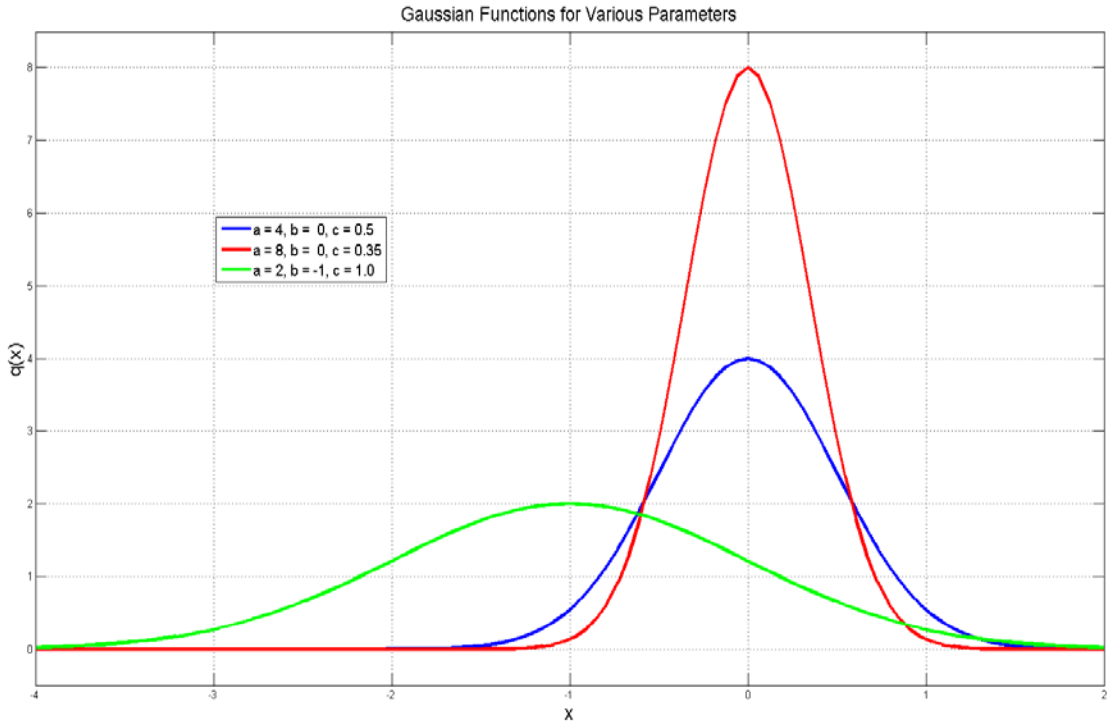

Figure 10. Gaussian Plot for Various Parameters

For our analysis of the IVP, we will set the domain to be $\{x \in \mathbb{R} \mid -1 \le x \le 1\}$. Additionally, we will impose periodic boundary conditions, such that the solutions at the right and left boundaries are equal to each other as the 1-D wave equation propagates to the right in time, for $0 \le t \le 1$, and $\Delta t = \sigma \frac{\Delta x}{c}$, where the time-step is a function determined by the Courant number, wave speed and grid spacing. For the following results, we will set the wave speed to be constant at $c = 2$, σ will vary depending on the stability of the numerical method and the grid spacing, Δx, will be determined by the number of grid points used to evaluate the solution for a particular time-step.

1. Explicit RK_2 Method

Let us first rewrite our IVP, Equation (2.9),

$$\frac{\partial q}{\partial t} = -c\frac{\partial q}{\partial x}, \qquad c > 0,$$

using Equations (2.5) and (3.7), where (2.5) is a second-order centered difference approximation for the continuous spatial derivative, and (3.7) is the RK_2 time-integration method. Here, we know that if

$$F(q_i^n) = -c\frac{\partial q_i^n}{\partial x} = -c\frac{q_{i+1}^n - q_{i-1}^n}{2\Delta x} + O(\Delta x^2), \quad \text{for} \quad c = 2,$$

then, substituting into (3.7) we find:

$$k_1 = \Delta t F(q_i^n) = -\frac{\Delta t}{\Delta x}\left(q_{i+1}^n - q_{i-1}^n\right)$$

$$\begin{aligned} k_2 = \Delta t F(q_i^n + k_1) &= \Delta t F\left(q_i^n - \frac{\Delta t}{\Delta x}\left(q_{i+1}^n - q_{i-1}^n\right)\right) \\ &= -\frac{\Delta t}{\Delta x}\left(q_{i+1}^n - \frac{\Delta t}{\Delta x}\left(q_{i+2}^n - q_i^n\right) - q_{i-1}^n + \frac{\Delta t}{\Delta x}\left(q_i^n - q_{i-2}^n\right)\right) \end{aligned}$$

$$\begin{aligned} q_i^{n+1} &= q_i^n + \frac{1}{2}\left(-\frac{\Delta t}{\Delta x}\left(q_{i+1}^n - q_{i-1}^n\right) - \frac{\Delta t}{\Delta x}\left(q_{i+1}^n - \frac{\Delta t}{\Delta x}\left(q_{i+2}^n - q_i^n\right) - q_{i-1}^n + \frac{\Delta t}{\Delta x}\left(q_i^n - q_{i-2}^n\right)\right)\right) \\ &= q_i^n - \frac{\Delta t}{\Delta x}\left(q_{i+1}^n - q_{i-1}^n\right) + \frac{\Delta t^2}{2\Delta x^2}\left(q_{i+2}^n - 2q_i^n + q_{i-2}^n\right). \end{aligned}$$

Therefore, we now have the following expression for RK_2 using a second-order centered difference stencil in space:

$$q_i^{n+1} = q_i^n - \frac{\Delta t}{\Delta x}\left(q_{i+1}^n - q_{i-1}^n\right) + \frac{\Delta t^2}{2\Delta x^2}\left(q_{i+2}^n - 2q_i^n + q_{i-2}^n\right). \tag{3.11}$$

If we fix our initial parameters to be $\sigma = 0.5$ and $\Delta x = 0.02$, then when we approximate Equation (2.9) using Equation (3.11), Figure 11 shows a graphical comparison between the exact solution and the numerical FD solution at times t = 0.025, 0.50, 1.0, and 4.0.

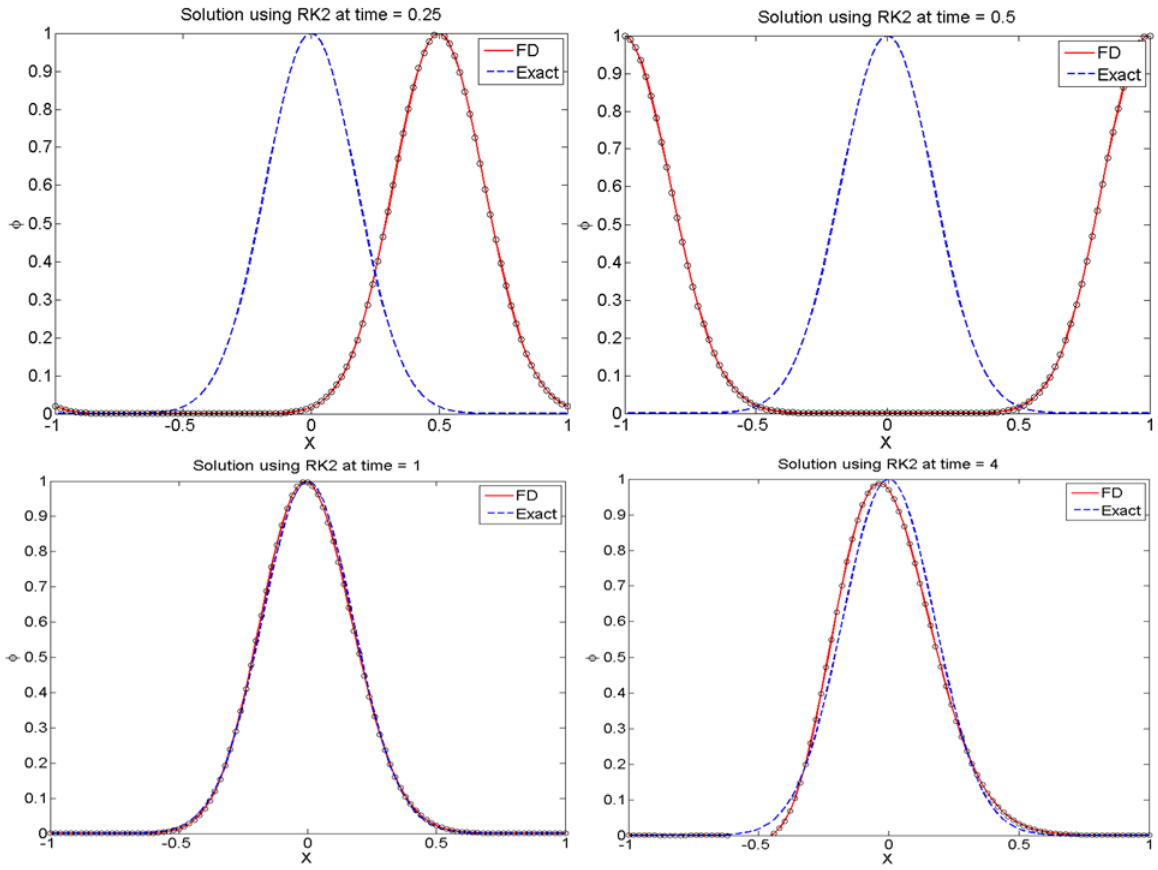

Figure 11. Numerical vs. Exact Solution for RK_2 using 2^{nd} Order CFD with $\Delta x = 0.02$

At time $t = 0$, the numerical solution (red curve) is set equal to the initial condition (blue curve), and as time increases, the initial condition (Gaussian function) begins to propagate to the right. From Figure 11 we can clearly see that after one revolution, where $t = 1$, the difference between the numerical solution and initial conidtion is approximately 0.0249, and at $t = 4$, the difference is 0.0986; where these norm values are computed using the L_2 norm, which was defined in Chapter II as

$$\|q\|_2 \equiv \left(\sum_{i=1}^{N} |q_i|^2 \right)^{1/2},$$

such that the difference between the numerical solution, q_i^n, and exact solution, ϕ_i^n, is

$$\left\| \phi_i^n - q_i^n \right\|_2 \equiv \left(\sum_{i=1}^{N} \left| \phi_i^n - q_i^n \right|^2 \right)^{1/2}, \quad \text{for} \quad n = 1, 2, 3, \ldots$$

It is easy to see from Figure 11, that the numerical solution, defined by the parameters above, quickly loses accuracy as time increases, such that both the dispersion and dissipation errors are evident. The dispersion error is a result of the numerical solution evolving slower in time compared to the real solution, while the dissipation error is the difference in the height of the numerical solution to that of the real solution. Therefore, since Δt is a function of the Courant number, wave speed and grid spacing, we have a few options to take in order to determine if we can achieve better results for solving (2.9) using (3.11): we can increase or decrease the number of grid points used to evaluate the solution (i.e., make Δx smaller / larger), modify the wave speed, choose a different Courant number, or a combination of the three, so long as we ensure that the stability of (3.11) is maintained.

Let us look at the results for only changing Δx. If we change the number of grid points from 100 to 400, then $\Delta x = 0.005$, and our time-step now becomes, $\Delta t = 0.00125$, compared to the previous time-step value of $\Delta t = 0.005$, for a fixed Courant number of $\sigma = 0.5$. Running the same simulation again with RK_2 for $0 \le t \le 4$, we notice from Figure 12 that the numerical solutions are more accurate when using $\Delta x = 0.005$, as opposed to $\Delta x = 0.02$ for $t \le 2$; however, for $t > 2$, the solution not only loses accuracy, but is unstable for the given parameters. Therefore, we see that although the initial computations using a finer grid provide better results for $t \le 2$, the end result is worse.

In fact, what we find is that the numerical solution is unstable for RK_2 using a fixed Courant value of $\sigma = 0.5$ and a Δx =0.005. Additionally, the computational cost also increased as we computed values at 400 points versus only 100 for each time-step. Initially, this did not make sense as I assumed using a finer grid spacing and smaller time step would produce better results. However, what we find is that as we refine the grid spacing smaller and smaller, the time step value needed to ensure stability must also get smaller.

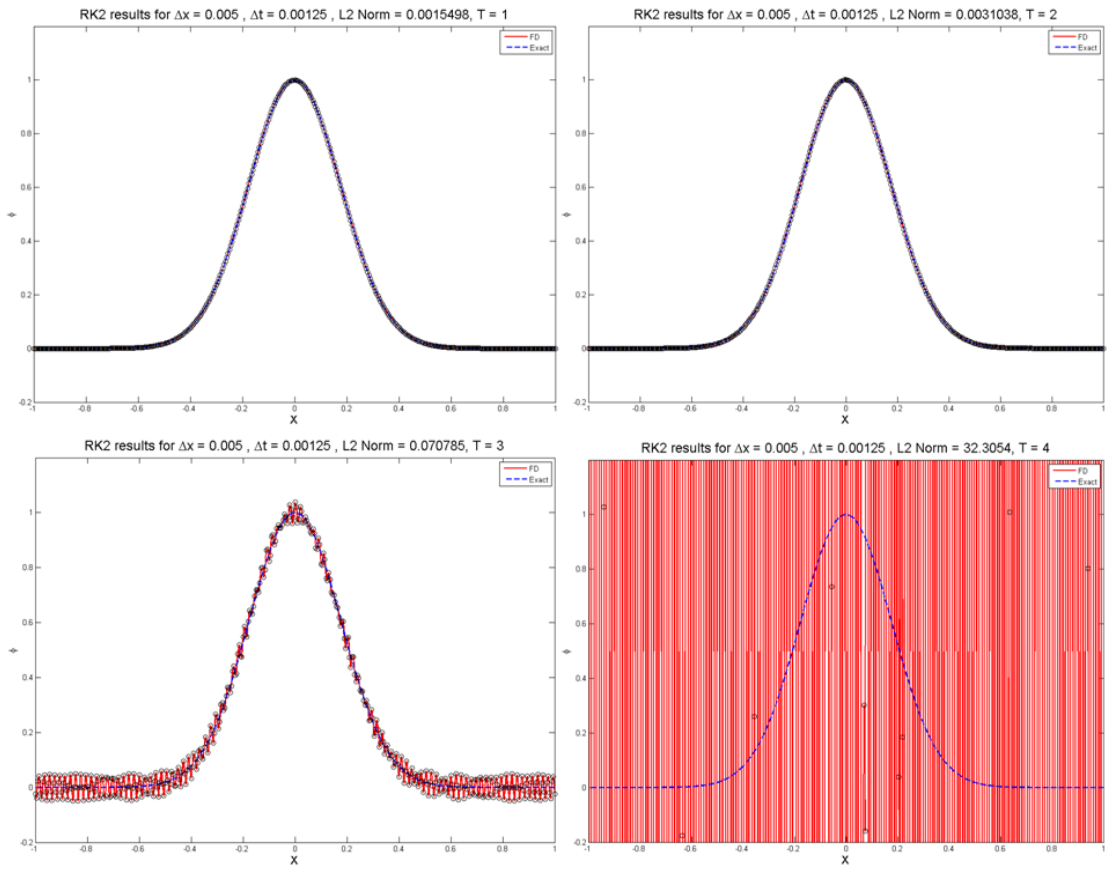

Figure 12. Numerical vs. Exact Solution for RK_2 using 2nd Order CFD with $\Delta x = 0.005$

Therefore, we need to look at the other parameters, namely σ, and find a solution that achieves the accuracy we desire, while simultaneously reducing the computational cost and ensuring stability. Furthermore, the results above show the importance of carrying out the simulation for increasing time. If we had stopped the simulation after only one or two revolutions, we would have assumed the solution for RK_2 using the finer grid for the parameters chosen was better, when in fact we have shown the opposite. Regardless, neither solution is desirable.

Instead, let us look at the solution to Equation (3.11) by fixing the grid spacing and only varying the Courant value. If we now plot the numerical solution using the Courant value and estimated error, we find that as the Courant value decreases, the estimated error of the solution decreases. These results are shown in Figure 13, and they confirm Equation (3.8), which shows that as the time-step is reduced by a factor of two,

the estimated error of the method is reduced by a factor of 2^p. Here, RK_2 is a second order method; therefore, as Δt is reduced by half, the estimated error should decrease by a factor of four. Although we are plotting against the Courant value, remember that

$$\sigma = c\frac{\Delta t}{\Delta x},$$

therefore, there is a direct correlation between σ and Δt, such that as σ gets smaller, Δt must also get smaller, given a fixed wave speed and grid spacing.

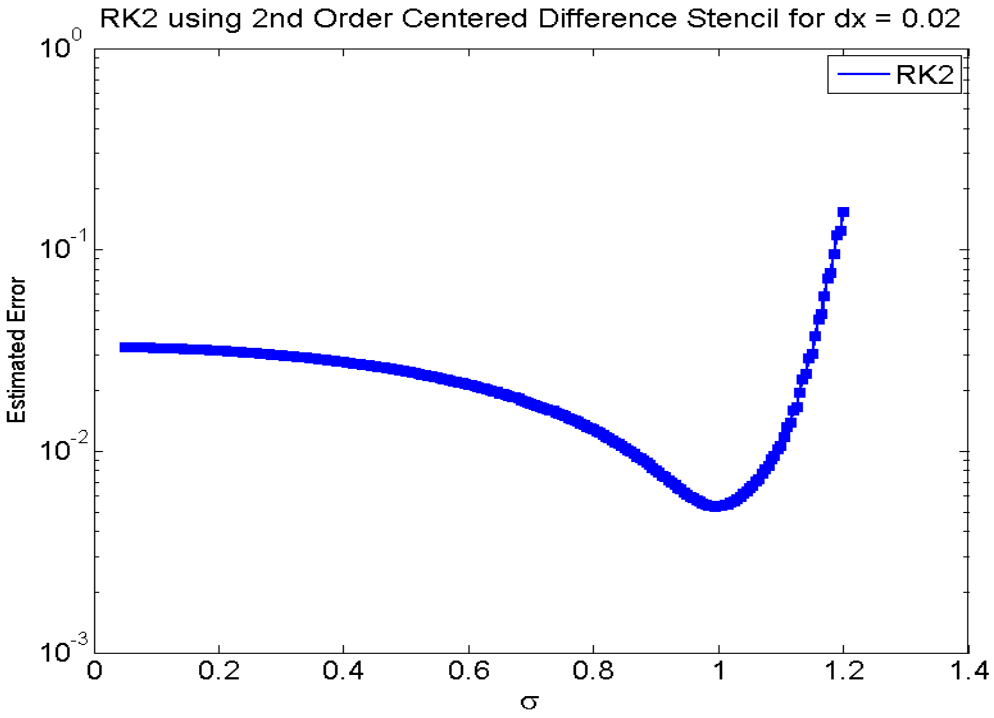

Figure 13. Estimated Error vs. Courant for RK_2 using 2nd Order CFD with Δx = 0.02

From Figure 13, we notice that the estimated error begins to increase for Courant values of $\sigma \leq 1$. What we find, is that this is not due to the time-integrator, but rather the spatial error introduced by our approximation of the spatial derivative. Here, we used Equation (2.5), which is the second-order, centered difference stencil. For our purposes, we are not as concerned with the spatial error, and only want to focus on the errors

produced from the various time-integration methods. Therefore, if we use a spatial discretization stencil with higher-order than the order of our time-integrators, we will hopefully be able to better analyze the solutions for decreasing time-steps or smaller Courant numbers. For that reason, let us now modify the numerical method so that we instead compute the spatial derivative using Equation (2.8),

$$\frac{\partial q_i^n}{\partial x} = \frac{q_{i-2}^n - 8q_{i-1}^n + 8q_{i+1}^n - q_{i+2}^n}{12\Delta x} + O(\Delta x^4),$$

which is the 4th order, centered difference stencil constructed in Appendix B.

If we again look at the estimated error versus Courant value plot using RK_2 in time and Equation (2.8) in space, then we find the following results, which are shown in Figure 14.

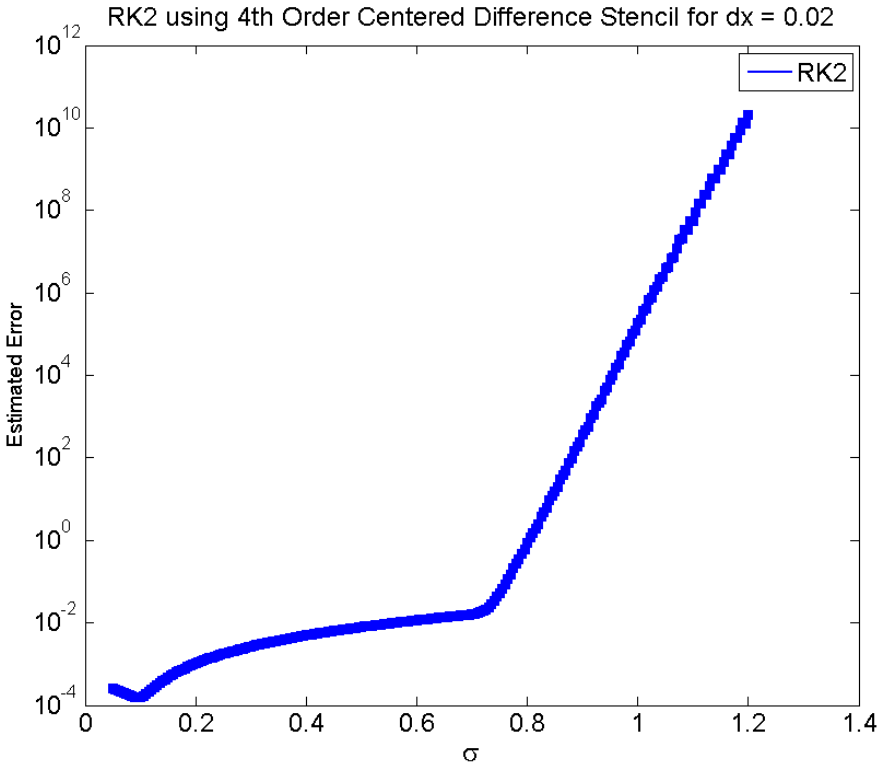

Figure 14. Estimated Error vs. Courant for RK_2 using 4th Order CFD with $\Delta x = 0.02$

From Figure 14, we now notice that our estimated error has improved drastically for decreasing values of σ; however, our solution becomes increasingly unstable for values of σ approximately larger than 0.75. Figure 14 clearly demonstrates the usefulness of higher-order spatial discretization methods when analyzing lower-order time integrators, such that we are better able to see the convergence rates of the time-integrator, as well as the estimated error. For example, both Figures 13 and 14 show the estimated error for the entire numerical method. Therefore, we notice that the spatial error dominates in Figure 13, so that we are unable to see the temporal errors; whereas in Figure 14, the spatial error is not readily seen until we reach a Courant value of approximately 0.1, thereby allowing us to more accurately analyze the temporal error for decreasing σ.

If we now vary both the grid spacing and Courant values, the results are shown in Figure 15. Here we find, that as the grid size decreases, the range of σ values must also decrease in order to maintain stability. This plot also demonstrates the trade-off between accuracy and efficiency, such that as Δx decreases, the more accurate the RK_2 solutions are to the exact solution; however, the number of computations per step must increase.

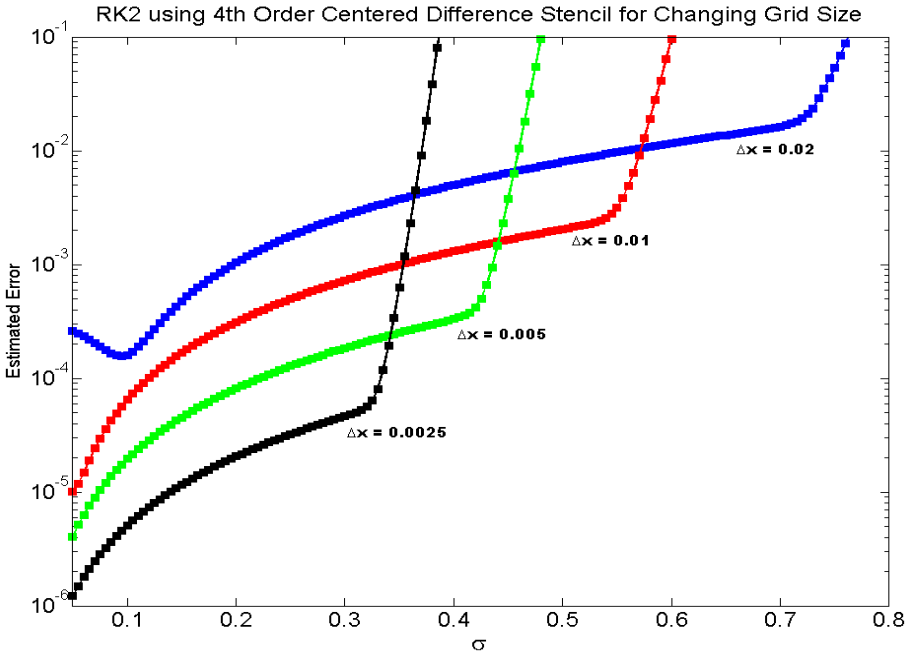

Figure 15. Error vs. Courant for Various Δx

Figure 15 is also useful in helping to better explain the previous RK_2 results, which were shown in Figures 11 and 12. Since we were using a fixed Courant value of 0.5 and only varying Δx, we notice that the blue curve supports the results shown in Figure 11, as we found that although the method gradually incurred both dissipation and dispersion errors, it was still stable. However, for the same Courant value of 0.5, we notice that the green curve represents the results shown in Figure 12, such that Figure 15 supports the fact that for $\Delta x = 0.005$, the numerical method is unstable.

Similar to Figure 15, it is also useful to plot the time-step, Δt, versus the estimated error of the numerical method. In Figure 16, we notice the areas along each curve, where the slopes are approximately equal to 2. It is within these regions that for a particular Courant value and spatial grid size, that the overall numerical method RK_2 is stable and 2^{nd} order accurate. The analytical proof that RK_2 is second-order accurate, regardless of the spatial discretization method, can be found in Appendix C.

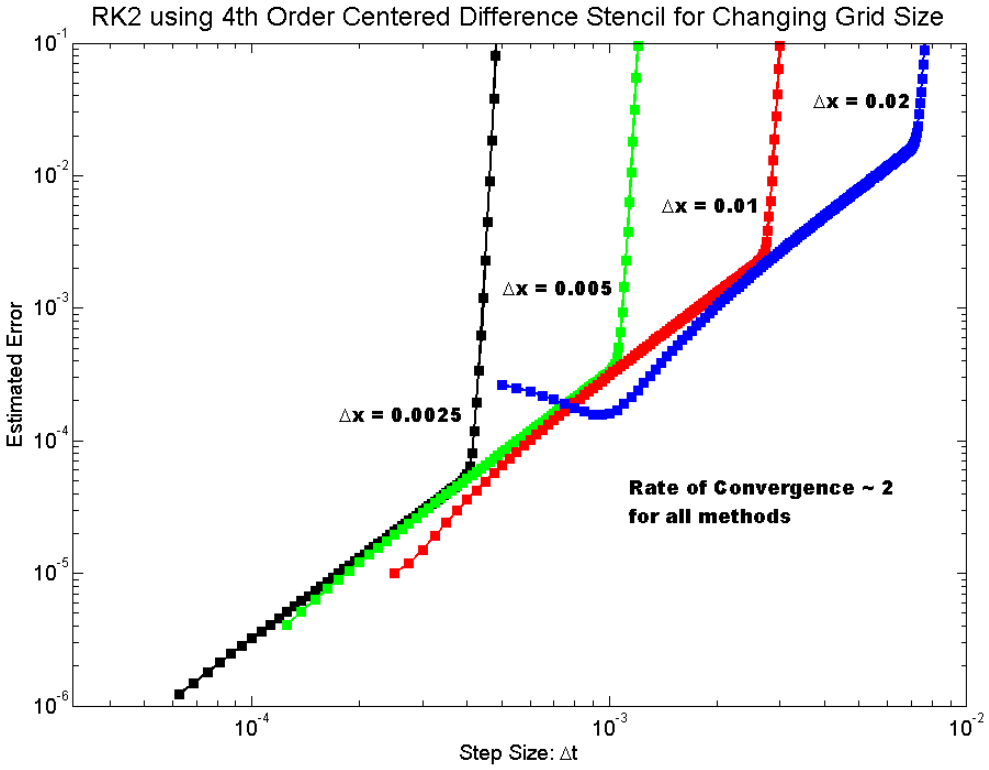

Figure 16. Error vs. Time-step for Various Δx

Figure 16 is also helpful, as it clearly shows the relationship between the time-step size and estimated error, as opposed to the Courant value. It is easily seen that for the given spatial grid sizes, the estimated error is reduced as Δx decreases; however, it is at the cost of a much smaller time-step, which requires more computational time.

In this chapter, we presented three different time-integration schemes: two multi-step (AB_2 and BDF_2), and one multi-stage (RK_2). Therefore, let us also present the results solving our IVP for all three methods.

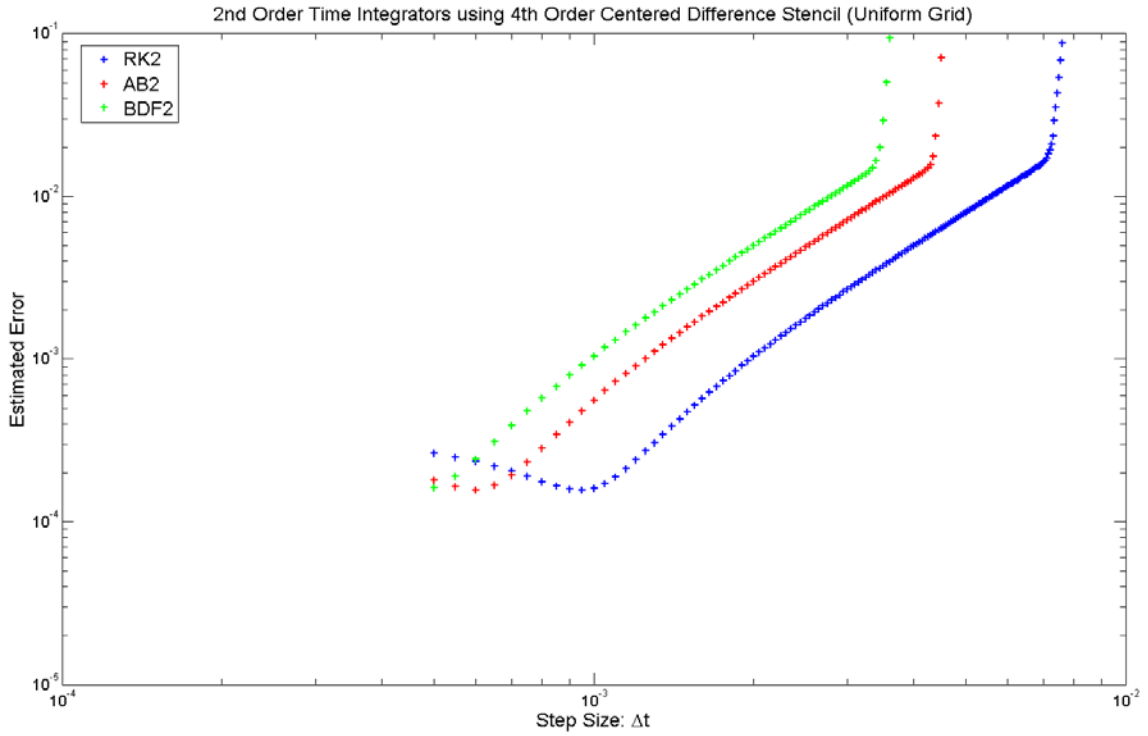

Figure 17. Error vs. Time-step for RK_2, AB_2, and BDF_2

From Figure 17, we notice that each of the three time-integrators converges to the actual solution at the expected rate of approximately two as each method is $O(\Delta t^2, \Delta x^4)$. We also notice that RK_2 is stable for larger time-step sizes, which supports the stability analysis plots shown in Figure 9. Therefore, given the above results, we will use RK_2 as the time-integration scheme of choice for the remainder of this thesis and in developing our multi-rate method in Chapter IV.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. MULTI-RATE METHODS

In this chapter, we focus on the development of a second-order multi-rate partitioned RK_2 method ($MPRK_2$), which uses a series of convex combinations of Euler steps [13]. To construct this method, we begin by introducing a non-uniform grid, where we must generalize the spatial derivative stencils developed in the previous chapters.

A. NON-UNIFORM GRIDS

In Chapters II and III, we solved our IVP using second and fourth-order accurate FD spatial discretization methods that were analyzed on a uniform grid. However, as we begin to construct our multi-rate time-integration method, we must now introduce a non-uniform mesh in both space and time. For now, we will simplify the problem to only non-uniformity in space.

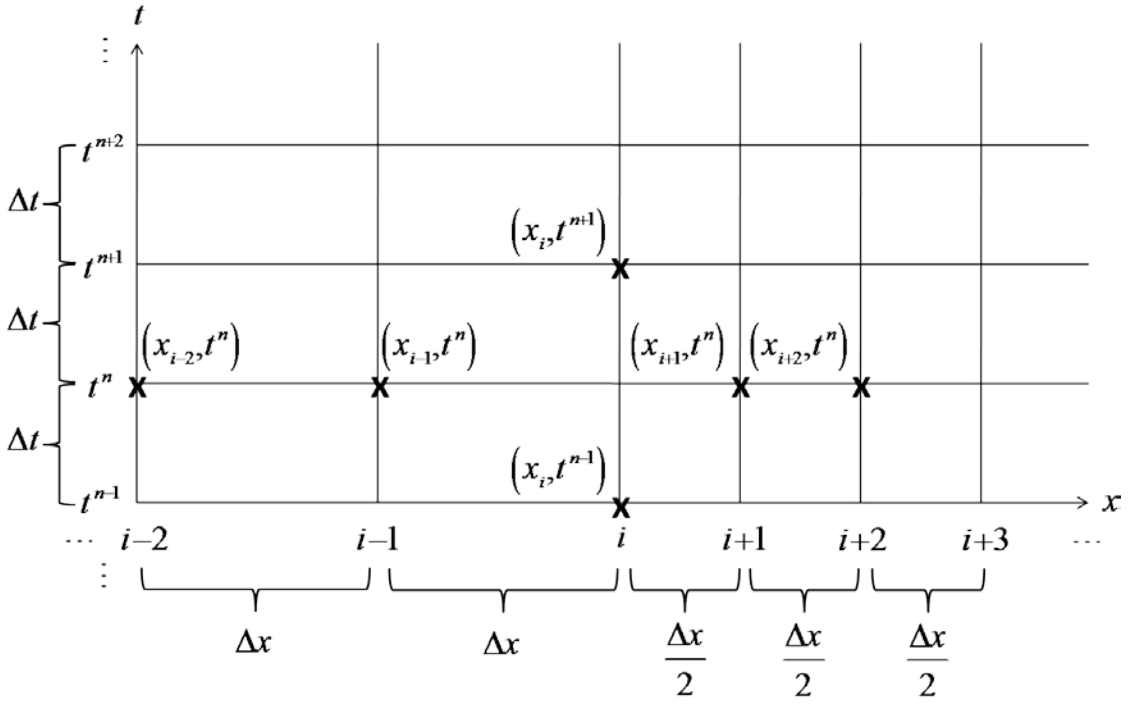

Figure 18. Non-uniform Grid in Spatial Domain

Using Figure 18, we notice that this grid is uniform in time for constant Δt, yet non-uniform in space. Here, we let every grid point to the left of x_i, for all time, be a distance of Δx apart, while all points to the right of x_i are a distance of $\Delta x/2$ apart. It is important to note, that in general, we can arbitrarily vary the distance between any two points within the non-uniform grid, and that Figure 18 is only one graphical example of how we may choose to define our grid space. Furthermore, this figure demonstrates that if we now want to build a discrete approximation to the spatial derivative of our IVP, then we will not be able to use the spatial stencils developed in Chapter II and Appendix B, as they all rely on a single Δx.

For example, Equations (2.5) and (2.8),

$$\frac{\partial q_i^n}{\partial x} = \frac{q_{i+1}^n - q_{i-1}^n}{2\Delta x} + O(\Delta x^2)$$

$$\frac{\partial q_i^n}{\partial x} = \frac{q_{i-2}^n - 8q_{i-1}^n + 8q_{i+1}^n - q_{i+2}^n}{12\Delta x} + O(\Delta x^4),$$

are the second and fourth-order centered difference stencils, respectively, which we used to approximate the spatial derivative of our IVP in Chapter III. Let us now look at how to construct a general expression for these stencils using the grid points at time, t^n, referenced in Figure 18. We will begin with the second-order centered difference stencil, such that we want to use the solutions at q_{i+1}^n, q_i^n and q_{i-1}^n, to approximate the first-order derivative of q_i^n with respect to *x*. Therefore, we now write the spatial discretization as a weighted sum of these solutions, where

$$\frac{\partial q_i^n}{\partial x} = \alpha_i q_{i+1}^n + \beta_i q_i^n + \gamma_i q_{i-1}^n \ . \tag{4.1}$$

We will begin with using TSE about q_i^n, such that,

$$q_{i+1}^n = q_i^n + \frac{\partial q_i^n}{\partial x}(x_{i+1} - x_i) + \frac{\partial^2 q_i^n}{\partial x^2}\frac{(x_{i+1} - x_i)^2}{2} + O\left[(x_{i+1} - x_i)^3\right]$$

$$q_{i-1}^n = q_i^n - \frac{\partial q_i^n}{\partial x}(x_i - x_{i-1}) + \frac{\partial^2 q_i^n}{\partial x^2}\frac{(x_i - x_{i-1})^2}{2} + O\left[(x_i - x_{i-1})^3\right] .$$

If we now substitute these expansions into Equation (4.1), then we achieve the following expression:

$$\frac{\partial q_i^n}{\partial x} = \alpha_i q_{i+1}^n + \beta_i q_i^n + \gamma_i q_{i-1}^n$$

$$\begin{aligned}\frac{\partial q_i^n}{\partial x} = {} & \alpha_i \left(q_i^n + \frac{\partial q_i^n}{\partial x}(x_{i+1} - x_i) + \frac{\partial^2 q_i^n}{\partial x^2}\frac{(x_{i+1} - x_i)^2}{2} + O\left[(x_{i+1} - x_i)^3\right] \right) \\ & + \beta_i q_i^n \\ & + \gamma_i \left(q_i^n - \frac{\partial q_i^n}{\partial x}(x_i - x_{i-1}) + \frac{\partial^2 q_i^n}{\partial x^2}\frac{(x_i - x_{i-1})^2}{2} + O\left[(x_i - x_{i-1})^3\right] \right) ,\end{aligned}$$

Since we want an expression for the second-order, centered difference stencil, we require the following equalities:

$$\begin{aligned} \alpha_i + \beta_i + \gamma_i &= 0 \\ \alpha_i (x_{i+1} - x_i) - \gamma_i (x_i - x_{i-1}) &= 1 \\ \alpha_i \frac{(x_{i+1} - x_i)^2}{2} + \gamma_i \frac{(x_i - x_{i-1})^2}{2} &= 0 , \end{aligned}$$

which can also be written as the following matrix problem,

$$\begin{pmatrix} 1 & 1 & 1 \\ \Delta_p & 0 & \Delta_m \\ \frac{(\Delta_p)^2}{2} & 0 & \frac{(\Delta_m)^2}{2} \end{pmatrix} \begin{pmatrix} \alpha_i \\ \beta_i \\ \gamma_i \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix},$$

where we let $\Delta_p = (x_{i+1} - x_i)$ and $\Delta_m = (x_i - x_{i-1})$. After solving for the coefficients using Cramer's rule, we find the following expressions:

$$\alpha_i = \frac{{\Delta_m}^2}{\Delta_p {\Delta_m}^2 + \Delta_m {\Delta_p}^2}, \qquad \beta_i = \frac{-\left({\Delta_m}^2 - {\Delta_p}^2\right)}{\Delta_p {\Delta_m}^2 + \Delta_m {\Delta_p}^2}, \qquad \gamma_i = \frac{-{\Delta_p}^2}{\Delta_p {\Delta_m}^2 + \Delta_m {\Delta_p}^2}.$$

Now, if we substitute the values for the coefficients α_i, β_i, and γ_i into Equation (4.1), we find a general expression for the spatial derivative, such that

$$\frac{\partial q_i^n}{\partial x} = \frac{{\Delta_m}^2 q_{i+1}^n - \left({\Delta_m}^2 - {\Delta_p}^2\right) q_i^n - {\Delta_p}^2 q_{i-1}^n}{\Delta_p {\Delta_m}^2 + \Delta_m {\Delta_p}^2} + O\left(\Delta_p \Delta_m\right), \tag{4.2}$$

which depends on the solution at three points with two distinct Δ parameters.

In order to verify that Equation (4.2) is the correct general formula for a second-order, centered difference stencil, then if we assume that both Δ parameters are equal (i.e., uniform grid), we can easily show that when $\Delta_p = \Delta x = \Delta_m$, then Equation (4.2) simplifies to Equation (2.5),

$$\frac{\partial q_i^n}{\partial x} = \frac{q_{i+1}^n - q_{i-1}^n}{2\Delta x} + O(\Delta x^2),$$

thereby proving that Equation (4.2) is a second-order accurate stencil.

Likewise, if we want to construct a general, fourth-order centered difference stencil, then following the same methodology as shown for developing Equation (4.2), we now find that the spatial derivative becomes

$$\frac{\partial q_i^n}{\partial x} = \alpha_i q_{i+2}^n + \beta_i q_{i+1}^n + \gamma_i q_i^n + \delta_i q_{i-1}^n + \eta_i q_{i-2}^n, \tag{4.3}$$

where after using TSE about q_i^n, we achieve the following matrix problem:

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ \Delta_{p2} & \Delta_{p1} & 0 & -\Delta_{m1} & -\Delta_{m2} \\ \frac{(\Delta_{p2})^2}{2} & \frac{(\Delta_{p1})^2}{2} & 0 & \frac{(\Delta_{m1})^2}{2} & \frac{(\Delta_{m2})^2}{2} \\ \frac{(\Delta_{p2})^3}{6} & \frac{(\Delta_{p1})^3}{6} & 0 & \frac{-(\Delta_{m1})^3}{6} & \frac{-(\Delta_{m2})^3}{6} \\ \frac{(\Delta_{p2})^4}{24} & \frac{(\Delta_{p1})^4}{24} & 0 & \frac{(\Delta_{m1})^4}{24} & \frac{(\Delta_{m2})^4}{24} \end{pmatrix} \begin{pmatrix} \alpha_i \\ \beta_i \\ \gamma_i \\ \delta_i \\ \eta_i \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix},$$

such that,

$$\Delta_{p2} = (x_{i+2} - x_i), \qquad \Delta_{p1} = (x_{i+1} - x_i)$$
$$\Delta_{m2} = (x_i - x_{i-2}), \qquad \Delta_{m1} = (x_i - x_{i-1}) \ .$$

Note that the general expression for the matrix above is

$$\begin{pmatrix} 1 & & & \cdots & & & 1 \\ \Delta_{i+\frac{k}{2}} & \cdots & \Delta_{i+1} & 0 & -\Delta_{i-1} & \cdots & -\Delta_{i-\frac{k}{2}} \\ \vdots & & & \vdots & & & \vdots \\ \frac{\left(\Delta_{i+\frac{k}{2}}\right)^k}{k!} & \cdots & \frac{(\Delta_{i+1})^k}{k!} & 0 & \frac{(-\Delta_{i-1})^k}{k!} & \cdots & \frac{\left(-\Delta_{i-\frac{k}{2}}\right)^k}{k!} \end{pmatrix},$$

where k is the order of the method, such that k is even.

After solving the matrix problem, we find the following expressions for the coefficients:

$$\alpha_i = -\frac{\Delta_{m1}\Delta_{m2}\Delta_{p1}}{\left(\Delta_{p2}+\Delta_{m2}\right)\left(\Delta_{p2}+\Delta_{m1}\right)\left(\Delta_{p2}-\Delta_{p1}\right)\Delta_{p2}} \tag{4.3.1}$$

$$\beta_i = \frac{\Delta_{m1}\Delta_{m2}\Delta_{p2}}{\left(\Delta_{m2}+\Delta_{p1}\right)\left(\Delta_{p2}-\Delta_{p1}\right)\left(\Delta_{m1}+\Delta_{p1}\right)\Delta_{p1}} \tag{4.3.2}$$

$$\gamma_i = \frac{\Delta_{m2}\left(\Delta_{p2}\left(\Delta_{p1}-\Delta_{m1}\right)-\Delta_{p1}\Delta_{m1}\right)+\Delta_{m1}\Delta_{p1}\Delta_{p2}}{\Delta_{m1}\Delta_{m2}\Delta_{p1}\Delta_{p2}} \tag{4.3.3}$$

$$\delta_i = -\frac{\Delta_{m2}\Delta_{p1}\Delta_{p2}}{\left(\Delta_{m2}-\Delta_{m1}\right)\left(\Delta_{p2}+\Delta_{m1}\right)\left(\Delta_{p1}+\Delta_{m1}\right)\Delta_{m1}} \tag{4.3.4}$$

$$\eta_i = \frac{\Delta_{m1}\Delta_{p1}\Delta_{p2}}{\Delta_{m2}\left[\Delta_{m2}{}^3+\Delta_{m2}{}^2\left(\Delta_{p1}+\Delta_{p2}-\Delta_{m1}\right)+\Delta_{m2}\left(\Delta_{p2}\left(\Delta_{p1}-\Delta_{m1}\right)-\Delta_{p1}\Delta_{m1}\right)-\Delta_{m1}\Delta_{p1}\Delta_{p2}\right]} \tag{4.3.5}$$

.

Therefore, if we substitute (4.3.1) - (4.3.5) into (4.3), we achieve the fourth-order, centered difference stencil we desired, such that

$$\frac{\partial q_i^n}{\partial x} = \alpha_i q_{i+2}^n + \beta_i q_{i+1}^n + \gamma_i q_i^n + \delta_i q_{i-1}^n + \eta_i q_{i-2}^n + O(\Delta_{m1}\Delta_{m2}\Delta_{p1}\Delta_{p2}) \,. \tag{4.4}$$

Notice, if $\Delta_{p1} = \Delta_{m1}$, $\Delta_{p2} = \Delta_{m2}$, and $2\Delta_{p1} = \Delta_{p2}$, then we can easily show that Equation (4.4) is equivalent to Equation (2.8), such that

$$\frac{\partial q_i^n}{\partial x} = \frac{q_{i-2}^n - 8q_{i-1}^n + 8q_{i+1}^n - q_{i+2}^n}{12\Delta x} + O(\Delta x^4) \,.$$

Let us return to our IVP , where we now solve for solutions using the non-uniform grid in Figure 18. If we use RK_2 as our time-integrator and Equation (4.2) as the spatial derivative, then we achieve the following results, which are shown in Figure 19:

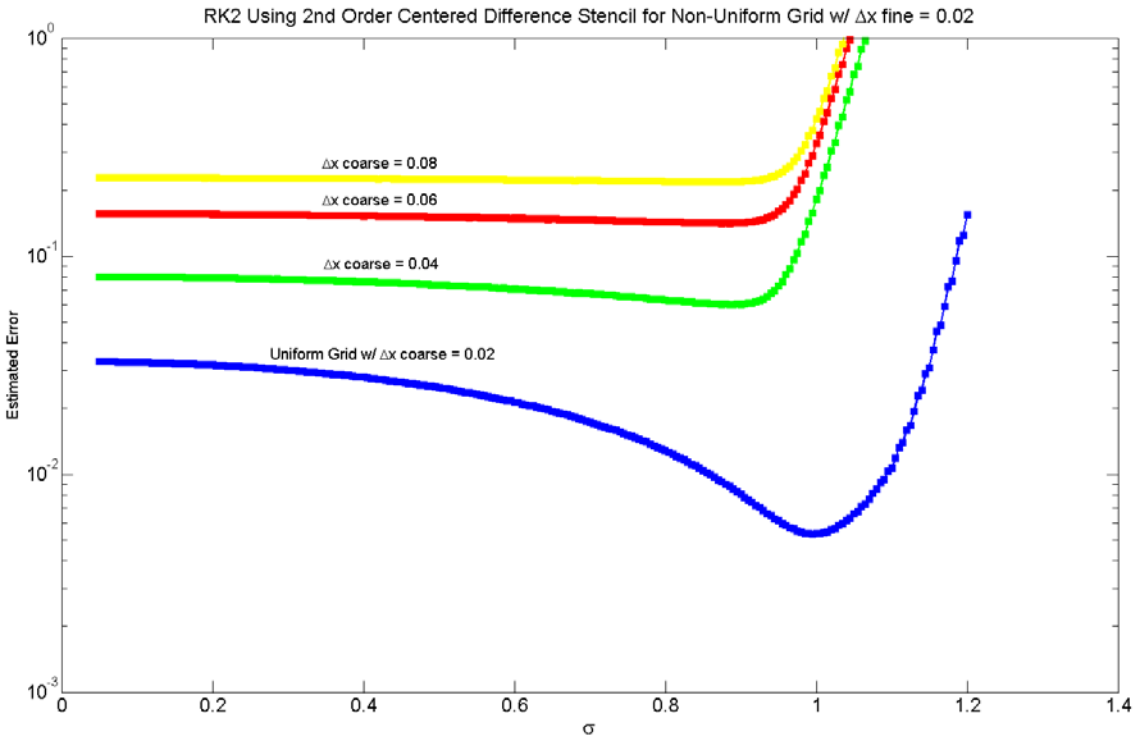

Figure 19. Non-uniform RK_2 Results: L_2 Error Norms vs. Courant Value using 2nd Order Centered Finite Difference Method

From Figure 18, we notice that our domain is partitioned into two sub-domains, such that the left sub-domain represents the course grid and the right sub-domain corresponds to the fine grid. Within Figure 19, we demonstrate that if we fix the fine grid, $\Delta x_{fine} = 0.02$, and vary Δx_{coarse} from 0.02 to 0.08, then as the width of Δx_{coarse} increases, the overall estimated error of the numerical method increases. We also notice that when we set $\Delta x_{coarse} = \Delta x_{fine}$, we return to a uniform grid, such that the errors are consistent with the results shown in Chapter III, specifically, Figure 13. This makes sense, as we should not expect to achieve better results using the non-uniform grid, compared to the uniform grid, provided the highest resolution, Δx_{fine}, within both grids is equivalent.

It is also important to analyze the L_2 error norms of both the uniform and non-uniform results, such that we should expect the same convergence rates. For example, we have shown for the uniform grid, that if our time-integration method is on the order of

$O(\Delta t^2)$, then as we reduce the step size by a factor of two, the estimated error of the method is reduced by a factor of 2^p, such that for RK_2, where $p = 2$, we know

$$O\left[\left(\frac{\Delta t}{2}\right)^2\right] = O\left(\frac{\Delta t^2}{4}\right).$$

However, when solving on the non-uniform grid, the time-step is now determined by the highest spatial resolution, in order to ensure stability of the overall method; therefore, Δt is a function of Δx_{fine}. Figure 20 displays the L_2 error norms for our uniform and non-uniform grid results using RK_2 and Equation (4.2). From this figure, it is easily seen that for the uniform grids, as Δx decreases by half, the estimated error of the method decreases by a factor of four; likewise, for the non-uniform grids, as Δx_{fine} decreases by half, where $\Delta x_{coarse} = 2\Delta x_{fine}$, the overall method also decreases by a factor of four. Figure 20 also verifies that the non-uniform grid results achieve larger errors than the uniform grid, when both methods use the same high resolution. For example, the non-uniform grid using $\Delta x_{fine} = 0.02$ and $\Delta x_{coarse} = 0.04$, achieves better results than the uniform grid with $\Delta x = 0.04$; however, is worse than the uniform grid with $\Delta x = 0.02$.

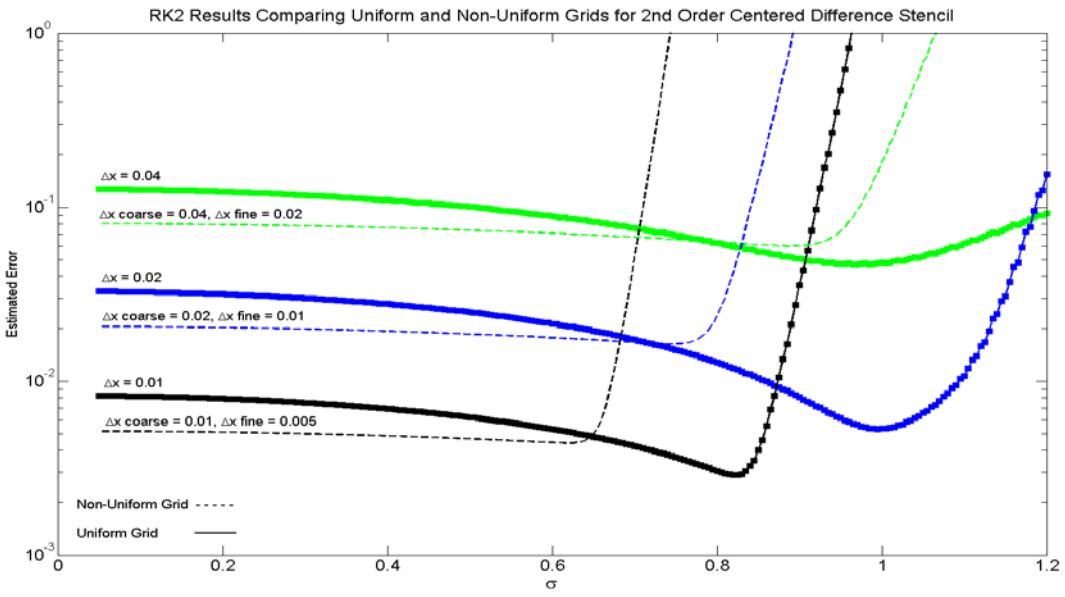

Figure 20. Uniform vs. Non-uniform RK_2 Results

It is also helpful to analyze the estimated errors of the uniform vs. non-uniform RK_2 results, using the degrees of freedom of the spatial mesh. Therefore, Figure 21 demonstrates that as the degrees of freedom of the numerical method increase, the estimated error decreases, such that the uniform grid will achieve better results than the non-uniform grid, given an equivalent number of points. This is the price one must pay for using the geometric flexibility of non-uniform grids, which must be used to yield more efficient solutions.

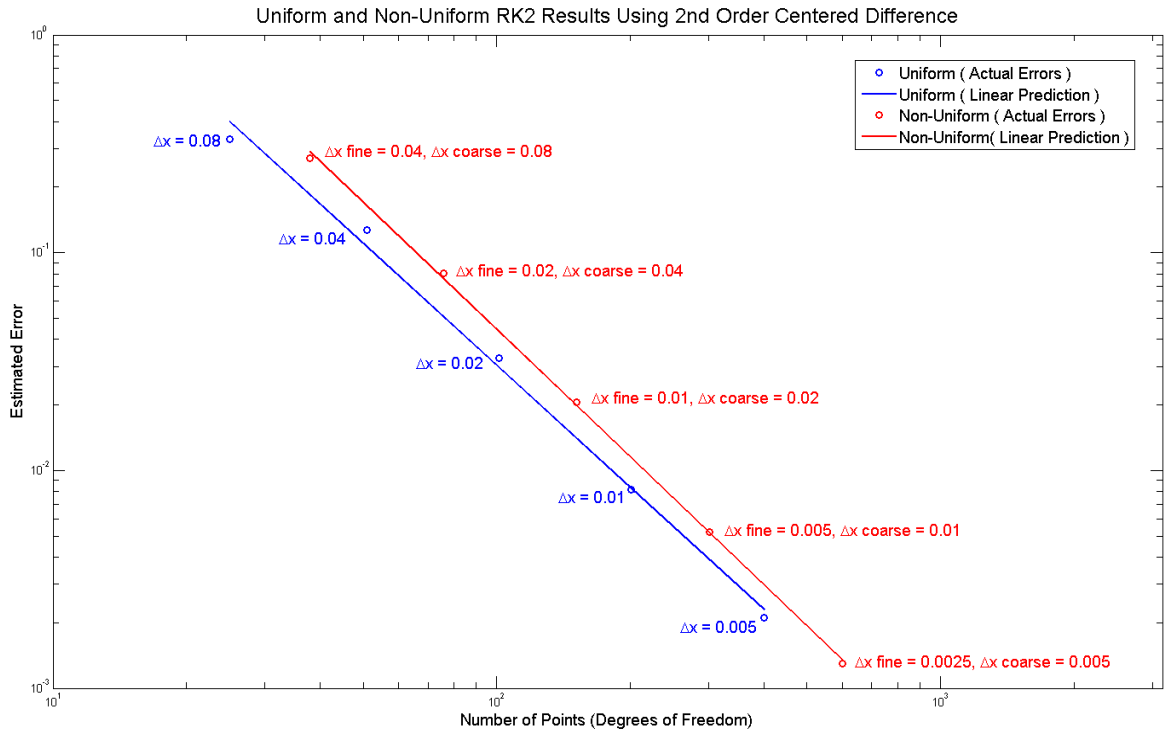

Figure 21. Estimated Error vs. Number of Points for Uniform and Non-uniform RK_2 Results

Although Figures 20 and 21 are helpful in comparing the uniform and non-uniform meshes, we notice that Figure 20 primarily shows the spatial error for the overall numerical method, which is on the order of $O(\Delta t^2, \Delta x^2)$. Therefore, in order to better analyze the temporal errors, we will now use RK_2 in combination with Equation (4.4), the general fourth-order centered difference stencil, to solve our IVP, such that the overall

numerical method is on the order of $O(\Delta t^2, \Delta x^4)$, when we set $\Delta x_{coarse} = \Delta x_{fine}$. Figure 22 displays these results, where we are now able to more accurately analyze the temporal errors for RK_2.

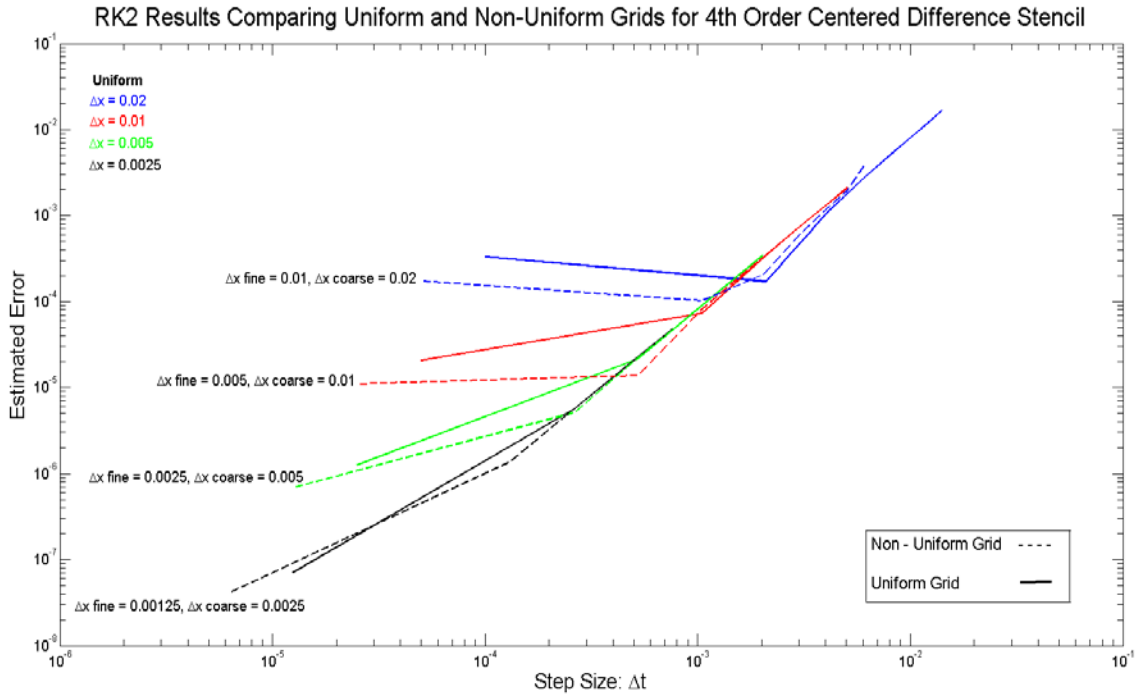

Figure 22. Uniform vs. Non-Uniform RK_2 Results using a 4th Order CFD Stencil

From Figure 22, we clearly see that the uniform results are equivalent to those found in Figure 16, and as we expected, introducing the fourth order centered difference stencil in space, has allowed us to view the temporal errors more effectively. We also notice from Figure 22, that our single-rate RK_2 method performs as we expected. For example, just as we found in Figure 20, the results above show that for a given non-uniform grid, the relative errors are greater than the uniform grid errors, when both methods use the same high resolution. Since we have validated that our general second and fourth-order spatial schemes, in conjunction with our single-rate RK_2 method are both consistent and accurate, let us now look at how to develop our multi-rate, partitioned, RK_2 time-integrator.

B. MULTI-RATE GRID DEVELOPMENT

In the previous section we looked at the single-rate RK_2 results on a non-uniform grid, such that the non-uniformity only existed within the spatial domain, as was shown in Figure 18. Let us now extend this non-conforming grid to both space and time, where Figure 23 displays a graphical representation for this non-uniformity.

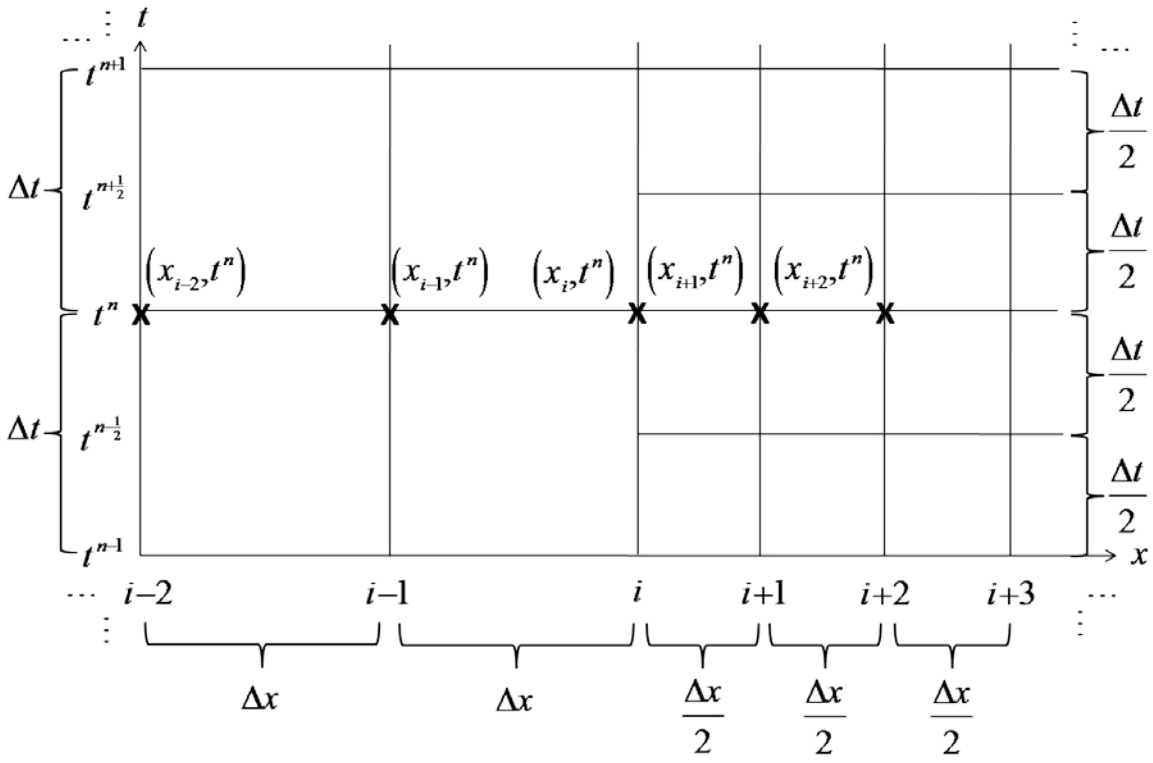

Figure 23. Non-conforming Grid in both Space and Time

From Figure 23, we now notice that the non-uniformity within the temporal domain will present a problem in constructing our finite difference representation of the continuous spatial derivative for specific points. Therefore, if we imagine that every grid point to the left of point "i" in Figure 23 is of dimension Δx, and that every grid point to the right of point "i" has dimension $\Delta x/2$, then it can be easily seen, that if we want to represent the spatial derivative of our IVP using the general fourth-order centered difference stencil developed in Chapter IV, that we will have no issues in constructing this stencil for grid points to the left of "i," or right of $i+1$, assuming that the grid extends infinitely in both

directions, thereby excluding any boundary conditions. However, we notice in Figure 24 that as time increases, we lack knowledge of information at every half time-step, that is necessary for building the fourth-order centered FD stencil for grid points "i" and $i+1$. These locations are indicated by red markers in Figure 24.

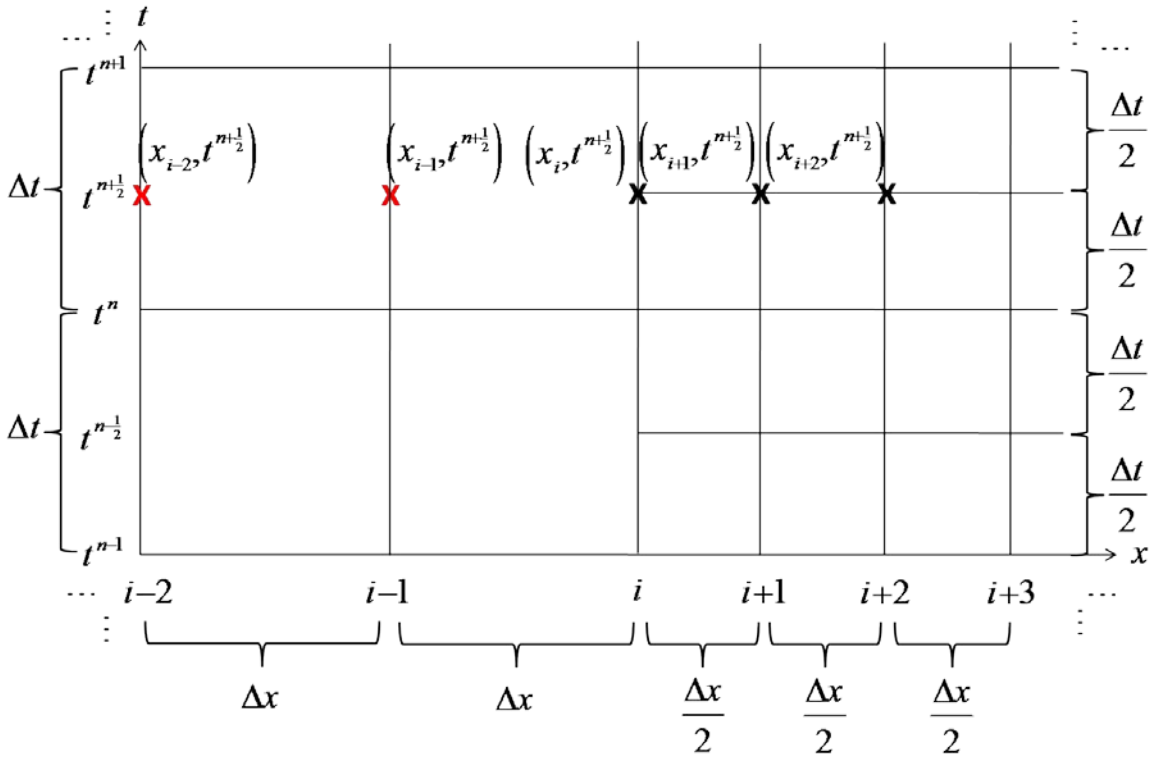

Figure 24. Lack of Information Needed to Build a 4th Order CFD Stencil

Let us now introduce some terminology, such that we will commonly refer to the coarse region within our spatial domain as the "slow region," and the fine region within the same domain as the "fast region," where the grid point located on the boundary between these two regions is an "interface point." Let us also assume from this point forward that when we refer to a FD stencil, that the reference stencil is the fourth-order centered FD stencil found in Equation (4.4) with weighted coefficients defined by Equations (4.3.1) – (4.3.5). From Figure 24, we notice that in order to represent the spatial derivative at grid points "i" and $i+1$, that we must make an approximation for the values at each half time-step where we do not have any information.

Using the information depicted in Figure 24, if we choose to let the values at grid points $x_{i-1}, t^{n+\frac{1}{2}}$ and $x_{i-2}, t^{n+\frac{1}{2}}$ be equal to x_{i-1}, t^{n} and x_{i-2}, t^{n} respectively, then it can be easily shown using TSE, that this approach will reduce the order of accuracy of the overall method. Likewise, if we choose to let these same points be equal to x_{i-1}, t^{n+1} and x_{i-2}, t^{n+1}, then this naïve approximation will also reduce the overall order of accuracy. However, if we know how to compute the information for grid points $i-1$ and $i-2$ at time levels t^{n} and t^{n+1}, then a better approach would be to take an average of the two values and use this information as placeholders for approximating our derivative, such that we can easily show using TSE, that this averaging approach will maintain the desired level of accuracy, such that our FD scheme will remain $O(\Delta x^4)$.

For our particular IVP, we are concerned with the boundary conditions, given that we enforce periodicity at these boundaries. Therefore, in order to simplify the problems associated with a non-conforming grid in both space and time at these boundaries, let us define a new grid that consists of three sub-domains (coarse / fine / coarse). Figure 25 shows a graphical representation of this grid, where the first and third sub-domains (slow regions) consist of same-sized elements, and the second sub-domain (fast region) consists of elements half the size of the slow regions.

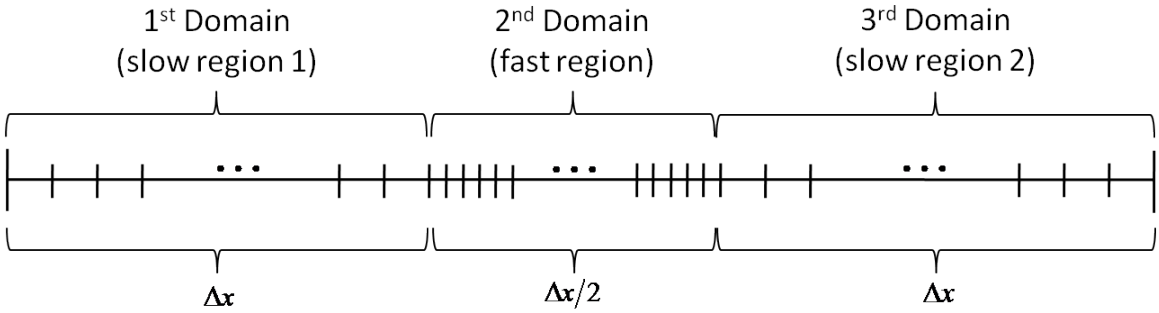

Figure 25. Non-conforming Grid with Three Sub-domains

This grid is different from the two sub-domain grid in Figure 24, in that we now have two interfaces which we will define as a slow/fast interface and a fast/slow interface, and can be seen in Figure 26. In addition, we also have what we will define as “buffer regions.”

These buffer regions exist at each interface, such that the size of a particular buffer region is dependent upon the grid points required to construct the FD stencil. First, we must construct slow and fast grids, which will be used to compute the slow and fast numerical solutions at a given time level. From Figure 26 we notice that the buffer regions are defined as the intersections of the slow and fast grids. Note, it is important to distinguish between the overall coordinate grid and the three sub-grids, as both slow grids in domains 1 and 2, each share grid points with the fast grid. Notice also that the fast and slow grids identified in Figure 26 are not the same as the fast and slow sub-domains shown in Figure 25.

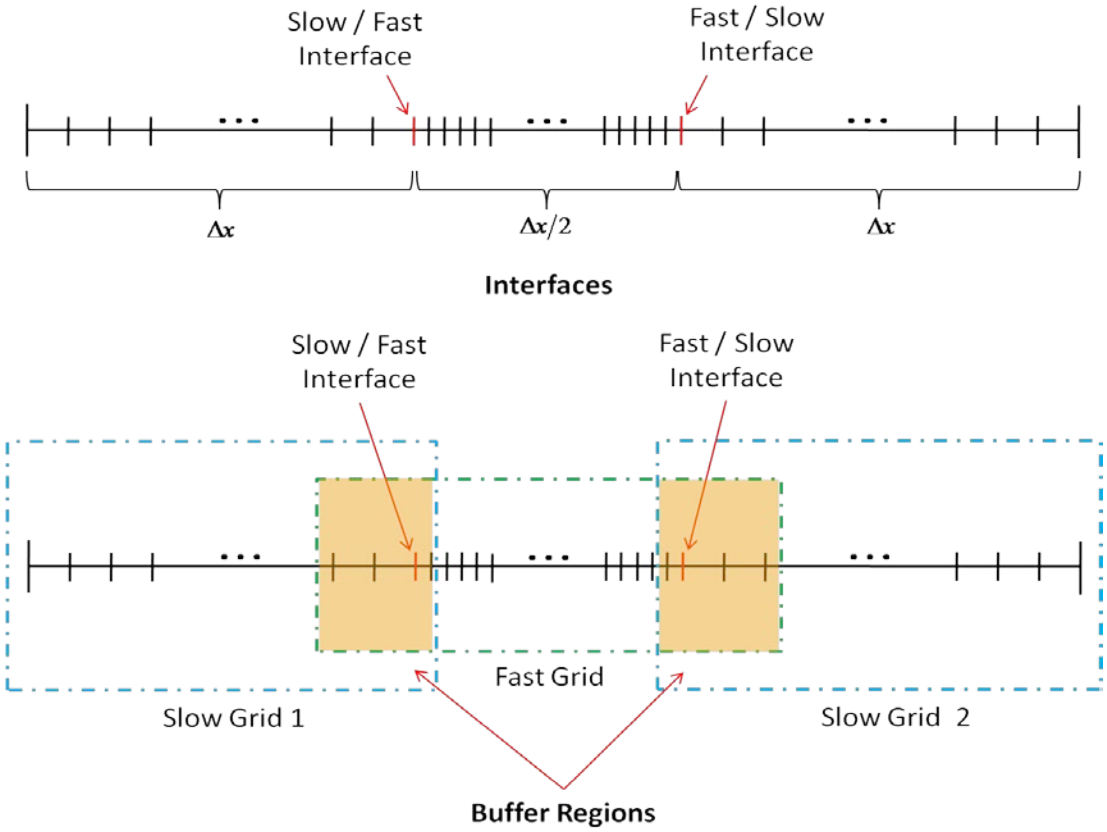

Figure 26. Interfaces and Buffer Regions for Non-Conforming Grid

Now that we have identified the difference between the slow and fast sub-domains, as well as the slow and fast grids, we must also define where our actual numerical solution will exist, such that in order to compute the entire solution at a given

time level, we require both the fast and slow solutions, where the individual solution at any given grid point "*i*," is located in one of three solution regions: purely fast, purely slow, or at an interface. However, the fast solution will take care of the interface solution. Let us refer to Figure 27 to see what we mean. From this figure, we notice that the fast solution is computed on the grid points within the fast sub-domain (including interface points) defined in Figures 25 and 26, whereas the slow solution is computed using the grid points within the two slow sub-domains minus the interface points.

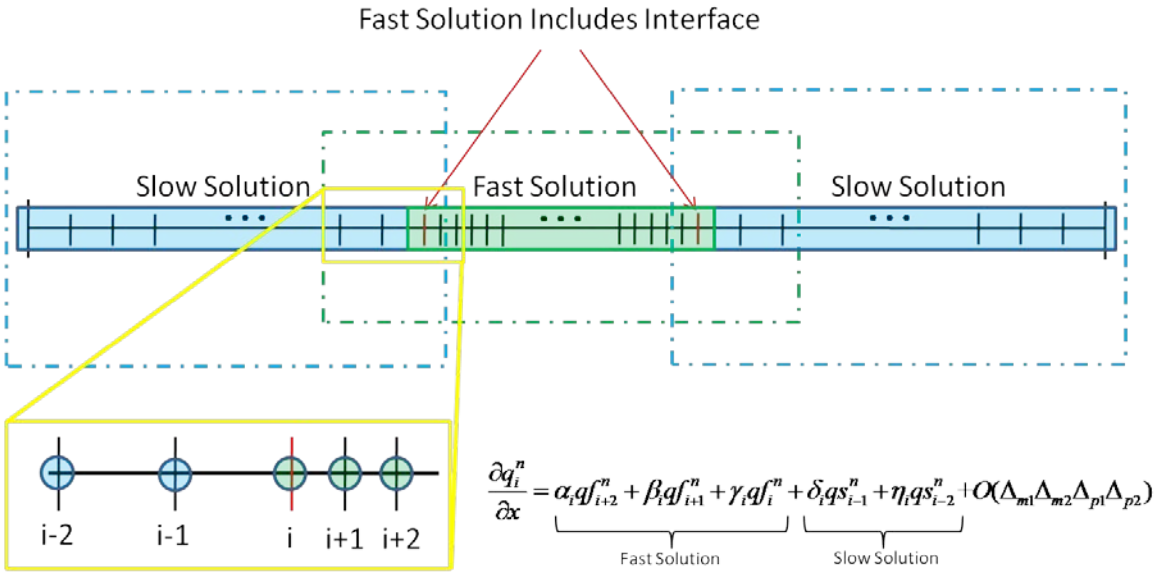

Figure 27. Computing a Fast or Slow Solution in the Buffer Region

For example, if we want to compute the FD fast solution at the first interface point located within the slow/fast buffer region, then we require information from the neighboring grid points, such that if the interface point is grid point "*i*," then we also need the values at grid points $i-2$, $i-1$, $i+1$, and $i+2$, such that the values at $i-2$, and $i-1$ will come from the slow solution and the values at $i+1$, and $i+2$ will come from the fast solution. However, as we saw in Figure 24, the slow solutions are computed on the slow domain, such that no information is stored at every half time-step. Therefore, let us now look at how to construct the multi-rate partitioned RK_2 ($MPRK_2$) time-integrator that will help us overcome this issue.

C. MULTI-RATE PARTITIONED RK_2 METHOD ($MPRK_2$)

As we begin to construct our $MPRK_2$ scheme, we want to ensure that all of the previous properties, such as consistency, stability and accuracy, as discussed in Chapter III also hold for this time-integration technique. According to [13], if we consider the second order, Runge-Kutta method, RK_2, defined by the Butcher tableau as

$$RK_2: \begin{cases} k_1 = \Delta t F(t^n, q^n) \\ k_2 = \Delta t F\left(t^n + \Delta t, q^n + k_1\right) \\ q^{n+1} = q^n + \frac{1}{2}\left(k_1 + k_2\right) \end{cases} \qquad \begin{array}{c|cc} 0 & 0 & 0 \\ 1 & 1 & 0 \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array} , \tag{4.5}$$

then using the following compact notation for explicit Euler steps,

$$\varepsilon(\Delta t, q) = q(t) + \Delta t F(t, q) , \tag{4.6}$$

as defined in [13], we can rewrite the above RK_2 method as a linear combination of Euler steps, such that we can guarantee that the above method will strongly preserve stability. This is known as a strong stability preserving (SSP) method, such that SSP time-integrators are methods that ensure a certain norm of the solution is bounded by the same norm of the previous time level, where

$$\left\| q^{n+1} \right\| \le \left\| q^n \right\| .$$

We notice that the RK_2 scheme (4.5) above is the same as Equation (3.9), where we have already proven in Appendix C that this time-integrator is indeed second order accurate, and from Chapter III, that the above RK_2 method, in conjunction with our fourth-order FD stencil, is consistent with the continuous PDE of our IVP. Furthermore, we analyzed the stability of this RK_2 method in the previous chapter using Von Neumann Stability Analysis. Therefore, if we now substitute a linear combination of Euler steps as defined by (4.6), into the above RK_2 method, we find from [13]:

$$
\begin{aligned}
q^{n+1} &= q^n + \frac{1}{2}\left(k_1 + k_2\right) , & q^{(1)} &= q^n + \Delta t F(t^n, q^n) = \varepsilon(\Delta t, q^n) , \\
&= \frac{1}{2} q^n + \frac{1}{2}\left(q^{(1)} + k_2\right) , & q^{(1^*)} &= q^{(1)} + \Delta t F(t^n, q^{(1)}) = \varepsilon(\Delta t, q^{(1)}) , \\
&= \frac{1}{2} q^n + \frac{1}{2} q^{(1^*)} , & q^{n+1} &= \frac{1}{2}\left(q^n + q^{(1^*)}\right) , \\
& & &= \frac{1}{2}\varepsilon(0, q^n) + \frac{1}{2}\varepsilon(\Delta t, q^{(1)}) ,
\end{aligned}
$$

where if the forward Euler method is SSP under its specific CFL time-step restrictions, then Shu and Osher [14] showed that higher-order methods constructed as linear combinations of forward Euler steps will also be SSP.

Now that we have rewritten the RK_2 method as

$$
q^{n+1} = \frac{1}{2}\varepsilon(0, q^n) + \frac{1}{2}\varepsilon(\Delta t, q^{(1)}) \tag{4.7}
$$

the intent is to use (4.7) to develop both a slow and fast RK_2 scheme to be used within the slow and fast domains of our three sub-domain grid developed in the previous section. We look again to the works by Constantinescu and Sandu [13], where they show how we can extend the above RK_2 base method to a second-order multi-rate partitioned Runge-Kutta (MPRK) scheme, where the MPRK scheme can be applied to multiple partitions, with "*m*" denoting the ratio between the time-steps associated with the fast and slow sub-domains on the same time level.

For our analysis, we consider only the case where our grid is refined at most once, such that the grid consists of at most two levels, which is seen in Figures 25 - 27. Additionally, for a two level grid, one level corresponds to the fast sub-domain, while the other level is associated to the slow sub-domain, such that the fast domain is refined at a two-to-one ratio. Therefore, we find that if the slow domain has elements of length Δx, than the fast domain will have length $\Delta x/2$. Furthermore, the ratio between the time-steps associated with the fast and slow sub-domains is also two-to-one, such that $m = 2$.

Let us now look at Figure 28, where we find the Butcher tableau for $m = 2$, such that the base method as defined by (4.5) can be rewritten for the slow and fast methods,

where the fast and slow methods weight coefficients are $1/mb$ repeated m times. Additionally, the slow method repeats the base methods stages m times with a time-step of Δt, while the fast method must perform m steps of the base method with a time-step of $\Delta t/m$. Furthermore, [13] shows how this technique of partitioning a base RK method can be extended from $m = 2$ to arbitrary m's.

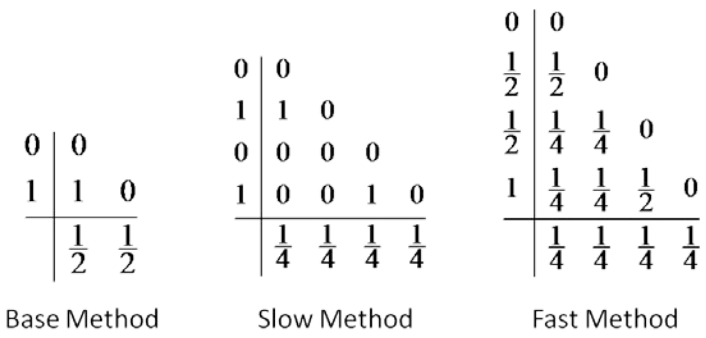

Figure 28. Butcher Tableau for RK_2 Method and its Associated Slow and Fast RK_2 Equivalents

We have already seen how we can rewrite the base RK_2 method using a linear combination of Euler steps. Therefore, let us now represent the slow and fast methods shown in Table 4 [13] using this same approach, where the reader is encouraged to go to [13] for proof of conservation and accuracy of these methods.

Looking at Table 4, we notice that each line in the table is part of an iterative process for computing the solutions within each sub-domain of the grid. For example, q_F^n and q_S^n are the initial solutions on both the slow and fast sub-domains, which initially correspond to the exact solution at time t^n. Therefore, once we initialize the solution vectors q_F^n and q_S^n, we have all the information needed to move to the second line in the table, and so on. Furthermore, the sequence in which each solution vector is computed, resolves the issue for how to compute the average value for grid points within the slow domain, that are needed when evaluating our fourth-order FD stencil in the buffer regions.

Fast Method	Slow Method (buffer region)	Base Method (slow region)
q_F^n	q_S^n	q_S^n
$q_F^{(1)} = \varepsilon_F(\frac{\Delta t}{2}, q_F^n, q_S^n)$	$q_S^{(1)} = \varepsilon_S(\Delta t, q_F^n, q_S^n)$	$q_S^{(1)} = \varepsilon(\Delta t, q_S^n)$
$q_F^{(1*)} = \varepsilon_F(\frac{\Delta t}{2}, q_F^{(1)}, q_S^{(1)})$	$q_S^{(1*)} = \varepsilon_S(\Delta t, q_F^{(1)}, q_S^{(1)})$	$q_S^{(1*)} = \varepsilon(\Delta t, q_S^{(1)})$
$q_F^{(2)} = \frac{1}{2}(q_F^n + q_F^{(1*)})$	$q_S^{(2)} = q_S^n$	
$q_F^{(3)} = \varepsilon_F(\frac{\Delta t}{2}, q_F^{(2)}, q_S^n)$	$q_S^{(3)} = \varepsilon_S(\Delta t, q_F^{(2)}, q_S^n)$	$q_S^{(3)} = q_S^{(1)}$
$q_F^{(3*)} = \varepsilon_F(\frac{\Delta t}{2}, q_F^{(3)}, q_S^{(3)})$	$q_S^{(3*)} = \varepsilon_S(\Delta t, q_F^{(3)}, q_S^{(3)})$	$q_S^{(3*)} = q_S^{(1*)}$
$q_F^{n+1} = \frac{1}{2}(q_F^{(2)} + q_F^{(3*)})$	$q_S^{n+1} = \frac{1}{2}q_S^n + \frac{1}{4}q_S^{(1*)} + \frac{1}{4}q_S^{(3*)}$	$q_S^{n+1} = \frac{1}{2}(q_S^n + q_S^{(1*)})$

Table 4. MPRK$_2$ Algorithm Used to Simultaneously Solve the IVP on Both the Fast and Slow Sub-domains

D. IMPLEMENTATION, RESULTS AND ANALYSIS

The implementation of this iterative time-integration process is quite simple. In fact, the most difficult step in advancing each slow and fast solution is within the spatial discretization scheme. Therefore, once the basic finite difference framework has been established, as was shown in Section B of this chapter, the individual computations of each Euler slow or Euler fast solution is trivial.

From Table 4, we notice that the base method is equivalent to Equation (4.7), and only requires knowledge of the solutions on the purely slow domain. However, the fast method performs four Euler-fast steps, where each step requires knowledge from the fast solution and the slow solution, if solving for grid points within the buffer region. Likewise, the slow method in the buffer region also requires information from the slow solution and fast solution. In other words, if a grid point lives in either a purely fast or purely slow region, then the base method will be used for the purely slow with only two Euler slow steps each of size Δt, and the purely fast with four Euler fast steps each of size $\Delta t/2$. It is only within the buffer region that the MPRK$_2$ method requires either four

Euler-fast or four Euler-slow steps, each requiring information from both the fast and slow solutions to compute the RHS FD stencil.

Let us now look at the results comparing the $MPRK_2$ method against the single-rate RK_2 for solving our 1D, first order, advection equation with constant wave speed and periodic boundary conditions for the smooth Gaussian function on the non-conforming three sub-domain grid as defined in Figure 25.

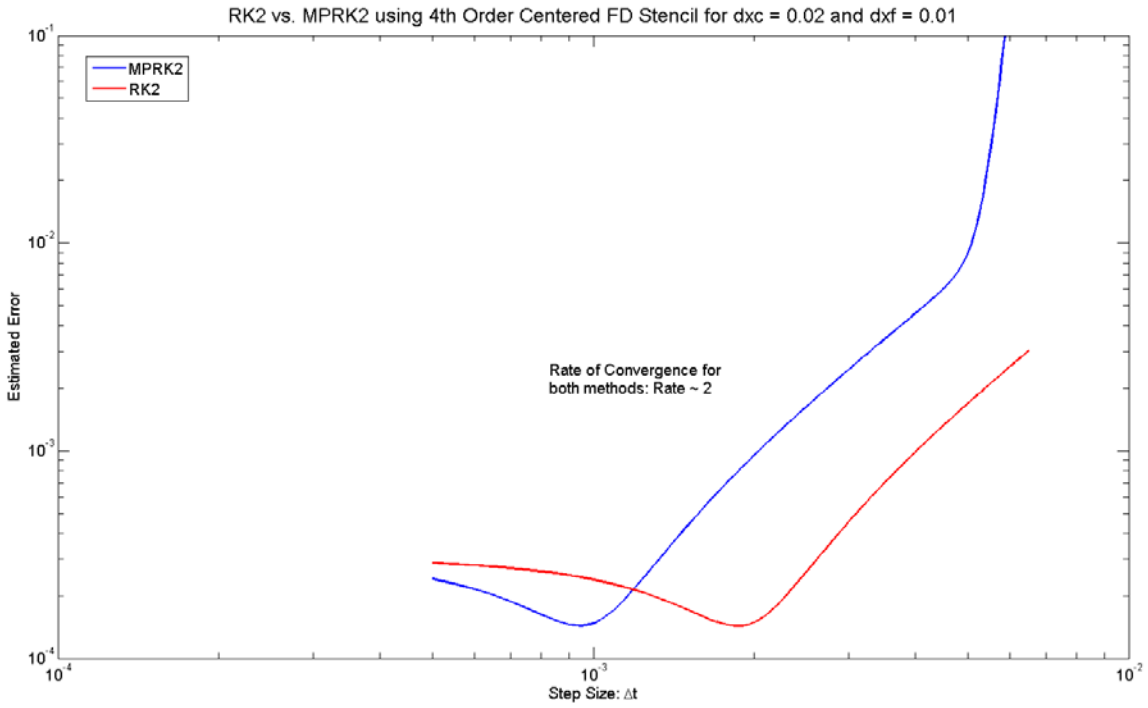

Figure 29. RK_2 vs. $MPRK_2$ Convergence Results using Approximately 100 Degrees of Freedom

From the above figure, we notice that both the $MPRK_2$ and RK_2 results converge on the order of $O(\Delta t^2)$ as $\Delta t \to 0$. However, for step-sizes Δt approximately less than10^{-3}, we begin to notice the spatial error from the fourth-order FD stencil. Furthermore, we also see that for a given time-step size, that the single-rate RK_2 method is clearly more accurate than the $MPRK_2$ for our particular choice of spatial element sizes Δx coarse $= 0.02$ and Δx fine $= 0.01$. In addition, Figure 30 shows a plot comparing the

efficiency of each method for these same parameters, such that for this size of problem, i.e., total number of grid points equal to 103, we notice that the single-rate RK_2 method is also more efficient.

Given these results, we now look to increase the number of points in the spatial domain, to determine if/when the multi-rate method will be more efficient. Before we do this, let us first look at the computational efficiency of each method, such that the reason for developing a multi-rate method in the first place was to have a more efficient numerical code capable of taking larger time-steps within the coarse regions and smaller time-steps within the fine regions of the spatial domain. As a result, the multi-rate method should indeed be more computationally efficient than a single-rate code that is restricted to taking a time-step dependant on the smallest grid size element in the spatial domain.

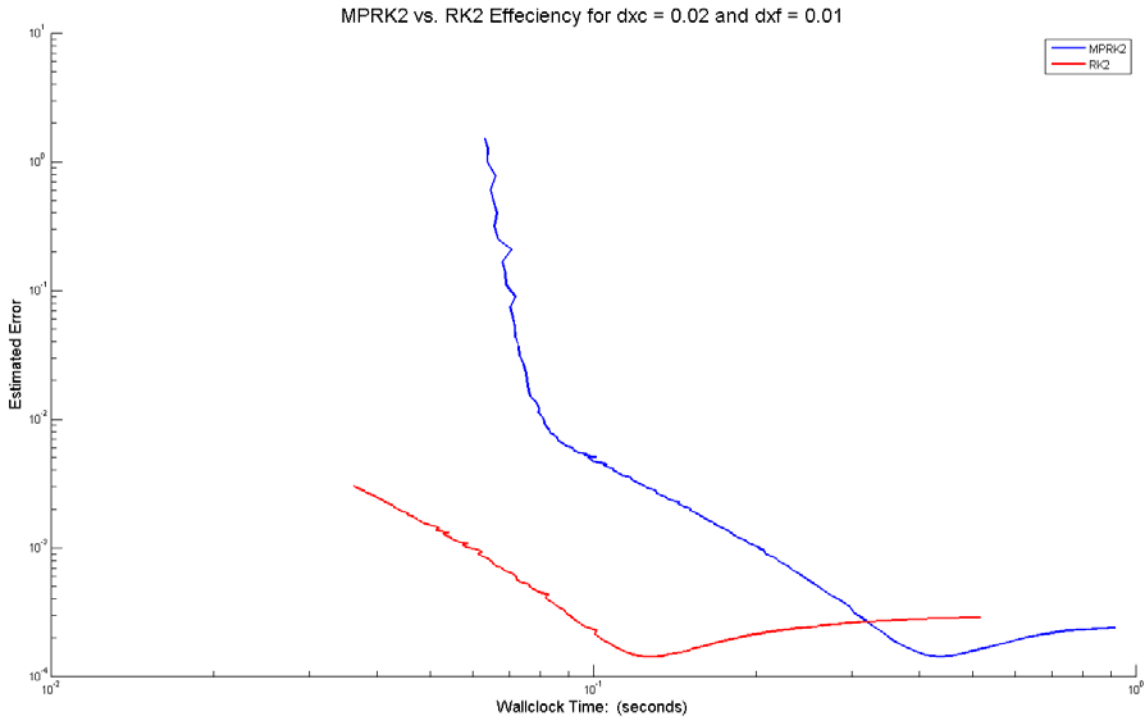

Figure 30. RK_2 vs. $MPRK_2$ Efficiency Results using Approximately 100 Degrees of Freedom

Both the single-rate and multi-rate methods are computed on the same grid, such that the multi-rate method solves for the purely fast and purely slow solutions using the same base method as the single-rate method, except for the purely slow solutions are capable of taking a time-step twice as large as the solution computed on the fast domain. Therefore, the only difference in computations is in the interface regions, which are typically small compared to the purely fast and purely slow regions. In other words, we truly have a multi-rate method, such that the slow regions do in fact use a step-size twice as large as the fast regions.

Analyzing the efficiency of our multi-rate time-integration method, shows that the speedup of this method is equivalent to the ratio of the total work done for the single-rate scheme with its restrictive time-step value ($\Delta t/2$), to the total computational work of the multi-rate scheme. Therefore, if we consider that the multi-rate method uses a step-size of $\Delta t/2$ on the fast domain, including interface points, with $N_F + N_I$ grid points, and a step-size of Δt on the slow domain with N_S grid points, then the multi-rate speedup is

$$S = \frac{mN_T}{m(N_F + N_I) + (m/2)N_S}\ , \tag{4.8}$$

where

N_T = total number of grid points $= N_F + N_S + N_I$,
m = number of fast steps,
N_F = number of purely fast grid points,
N_I = number of interface grid points,
N_S = number of purely slow grid points.

Furthermore, we find that since the total number of interface points $N_I \ll \min(N_S, N_F)$, then for $m = 4$, Equation (4.8) simplifies to

$$S \approx \frac{2N_T}{2N_F + N_S}\ . \tag{4.9}$$

Additionally, taking the limit of S as $N_S \to N_T$, such that $N_F \to 0$, then we find that

$$\lim_{\substack{N_S \to N_T \\ N_F \to 0}} S = 2, \tag{4.10}$$

where Equation (4.10) shows that the maximum theoretical speedup of our $MPRK_2$ solution is equal to 2, regardless of the value of *m*. This makes sense as we have assumed that there is only one level of refinement at a two-to-one ratio.

Below, Figure 31 shows the $MPRK_2$ vs. RK_2 wallclock time results versus time-step size for Δx coarse $= 0.02$ and Δx fine $= 0.01$. Using (4.8), we find that the theoretical speedup should by approximately 4 percent for $N_T = 103$ grid points. However, from this figure, we notice that there was no computational speedup. In fact, the single-rate method was still more efficient for the above parameters. Note, that the difference between the theoretical speedup and lack of computational speedup (in this case specifically) is most likely attributed to other coding inefficiencies, and not the $MPRK_2$ scheme. Therefore, we expect to achieve close to the theoretical speedup as we increase the Degrees of Freedom (DoF).

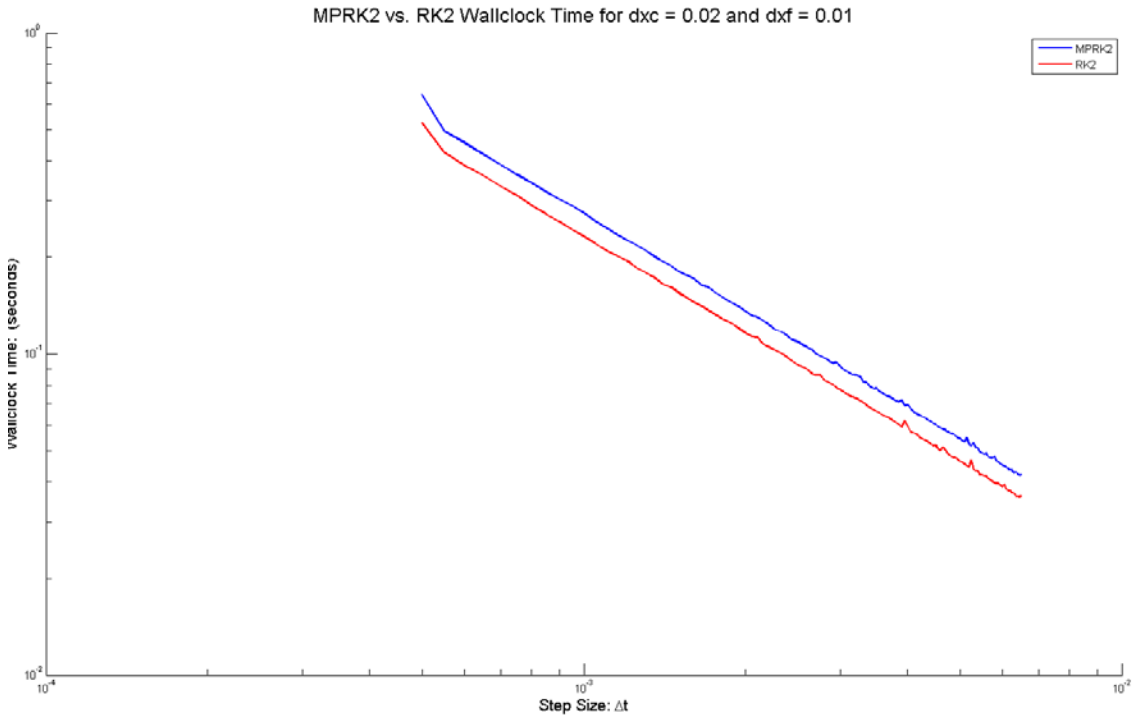

Figure 31. RK_2 vs. $MPRK_2$ Effeciency Results (~100 DoF)

If we now consider Δx coarse $= 0.00125$ and Δx fine $= 0.000625$, then the total number of grid points is approximately 1600. Figure 32 displays the convergence results for these new parameters, such that we find the single-rate RK_2 method is still more accurate for a given time-step size; however, Figure 33 shows that the $MPRK_2$ method is now more efficient for any given time-step.

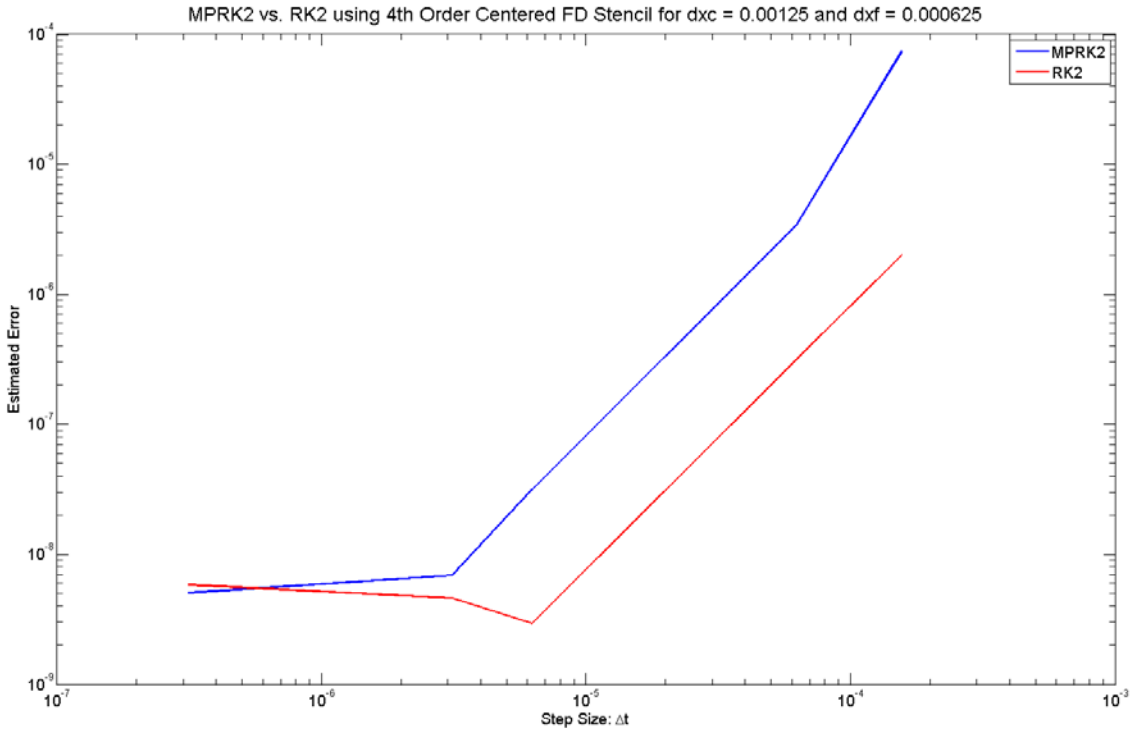

Figure 32. RK_2 vs. $MPRK_2$ Convergence Results using Approximately 1600 DoF

Comparing the theoretical versus computational speedup for the newly defined parameters, Δx coarse $= 0.00125$ and Δx fine $= 0.000625$, then using Equation (4.9), we notice the following results where $N_T = 1,603$, $N_F = 5$, and $N_S = 1,596$:

$$S \approx \frac{2N_T}{2N_F + N_S} \approx 1.996.$$

This shows that for the problem above, which has approximately 1,600 grid points, that the theoretical speedup should be approximately 2 (100% speedup). Using the data from

Figure 33, we find that the computational speedup we achieved using the $MPRK_2$ scheme is approximately 1.67 (67% speedup). These results are significant, and are exactly what we hoped to achieve. In general, we have proven that the multi-rate method formulated in Table 4, is indeed more efficient than the single-rate RK_2. Therefore, given that both the RK_2 and $MPRK_2$ time-integration schemes are on the order of $O(\Delta t^2)$, the primary consideration one must make between the two methods is whether one desires a more accurate, but less efficient approximation, or a slightly less accurate, yet more efficient approximation to the IVP. In addition, let us also look at the estimated errors of each method versus the computational time, such that Figure 34 again shows that the single-rate RK_2 method is still more efficient overall in reaching a specific level of accuracy, than the multi-rate method.

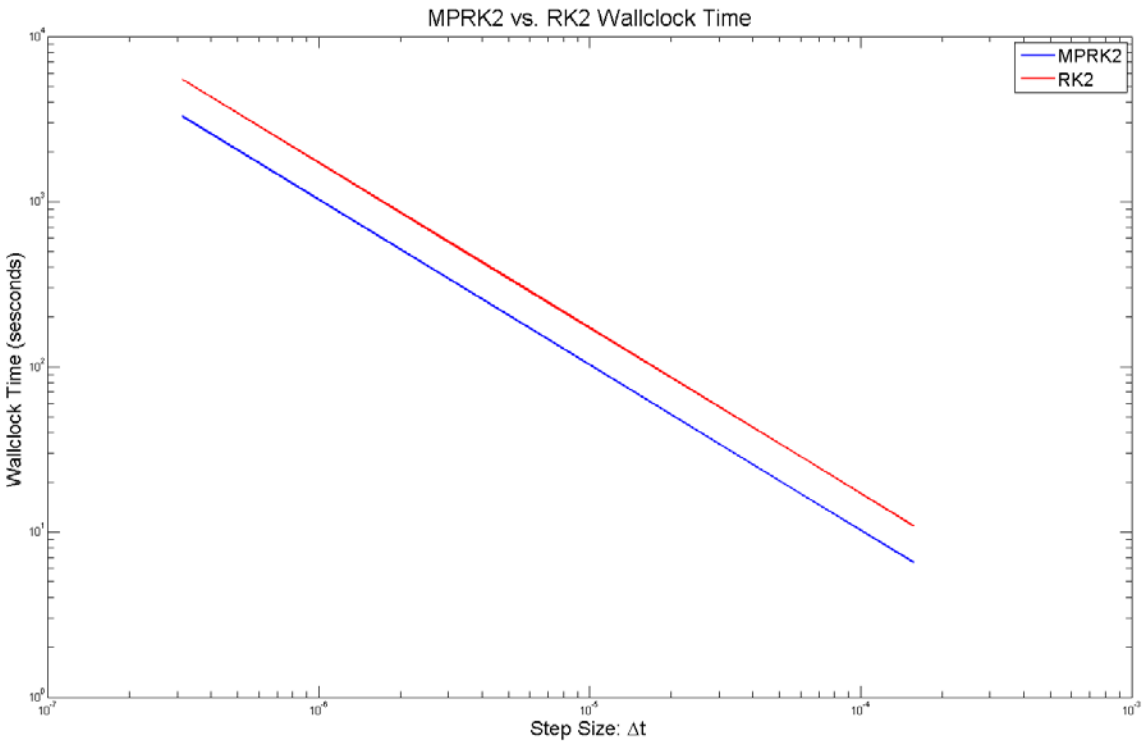

Figure 33. RK_2 vs. $MPRK_2$ Efficiency Results using Approximately 1600 DoF

Although Figure 33 demonstrates that for any given time-step the multi-rate method was computationally more efficient, the results in Figure 34 were initially

discouraging. However, it can be easily shown that the increased accuracy of the single-rate method is attributed to the fixed non-uniform grid, whereas the multi-rate method loses accuracy due to this same spatial mesh. In other words, the single-rate method should indeed be more accurate as it solves the IVP everywhere on the fixed non-uniform grid with a single time-step associated with the smallest element within the spatial domain; whereas, the multi-rate method uses a time-step twice as large within the entire slow sub-domain. From the results above, we see that the slow sub-domain consisted of 1,596 grid points out of a total 1,603 points. Although the initial grid was refined underneath the bell of the Gaussian curve, the right moving wave was rarely inside the fast region; therefore, the numerical solution lost accuracy as the wave moved from the fast region into the larger slow region.

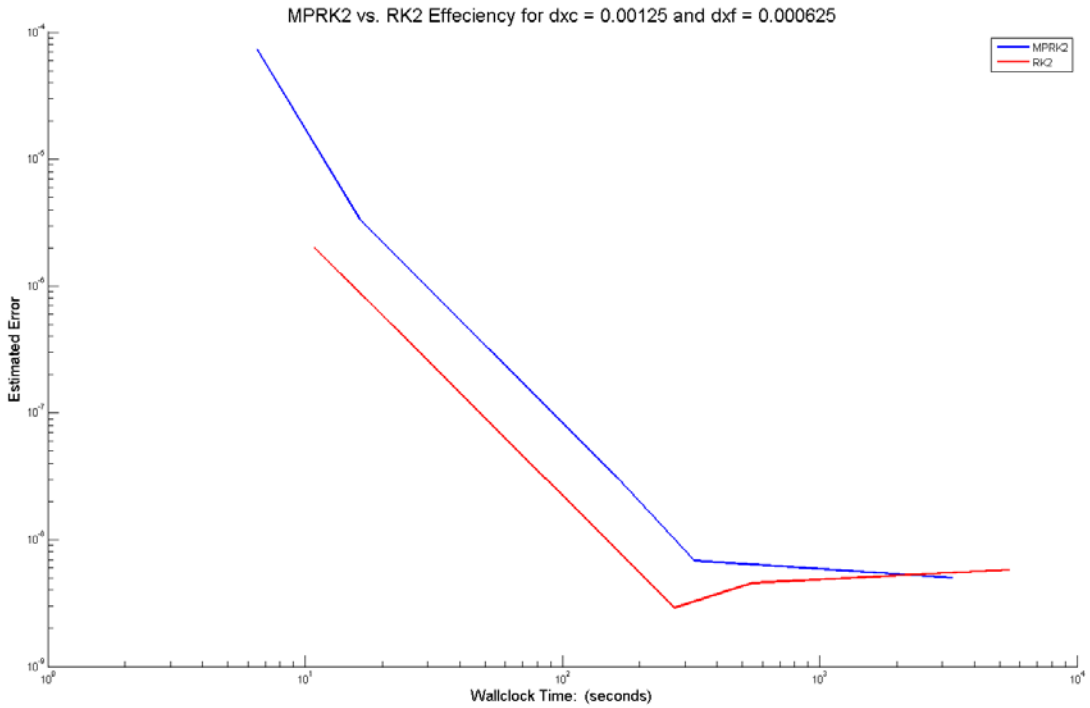

Figure 34. RK_2 vs. $MPRK_2$ Efficiency Results using Approximately 1600 DoF

V. SUMMARY AND CONCLUSIONS

Throughout this thesis we have shown how to use the method of finite differences, as well as a few explicit time-integration schemes, in order to construct a discretized form of our continuous, hyperbolic initial value problem, such that we were able to successfully solve this IVP numerically. In addition, we demonstrated how to use a linear combination of strong stability preserving, explicit forward Euler steps, in order to develop a 2^{nd} order Runge-Kutta, multi-rate time-integration scheme, where the results from Chapter IV demonstrate that this $MPRK_2$ method is indeed more computationally efficient than its equivalent single-rate RK_2 method for any given time-step size.

However, there is a great deal of research that must still be accomplished. Therefore, I recommend the following areas for future research:

1) Incorporate an adaptive mesh refinement method in conjunction with the multi-rate scheme.
2) Increase the dimensionality of the problem.
3) Vary the initial and boundary conditions.
4) Develop other multi-rate time-integration schemes using AB_2 or BDF_2.

Although the one-dimensional efficiency results comparing time-step sizes versus wall-clock time were successful, as shown in Figure 33, it would be beneficial to see what impact incorporating recommendation (1) would have on improving the multi-rate accuracy results displayed in Figure 34. If the refined sub-domain within the spatial grid, Figure 25, were to adapt in time to stay under the bell of the Gaussian curve, then we should expect that the $MPRK_2$ method would not only be more efficient for a given time-step size, but also achieve a particular level of accuracy faster than the single-rate RK_2 method.

The second recommendation is to increase the dimensionality of the problem being solved. If we are able to improve both the efficiency and accuracy of our numerical solution in only one-dimension, then it follows that an adaptive multi-rate time-

integration scheme should also improve the speedup in higher dimensions. In addition to this recommendation, it would also be beneficial to incorporate other initial conditions, such that we should analyze how well the multi-rate method is capable of solving the same test case IVP with either a square wave or another Gaussian wave with source.

Additionally, future analysis should be done on other PDE's or systems of PDE's. Although the 1^{st} order advection equation was a good test case PDE, the linearized shallow water equations, Burgers' equation, or even Maxwell's equations, each allow for a more thorough analysis of how well the $MPRK_2$ time-integration scheme compares to any other single-rate method. Furthermore, although it is important to analyze a particular multi-rate approach with its equivalent single-rate method, as was shown in this thesis, other multi-rate time-integration schemes should also be developed in order to compare each multi-rate approach against another.

APPENDIX A: TSE OF EQUATION 2.1

We begin the proof for Equation 2.1 by using Taylor series expansion (TSE) to expand each grid point about the point q_i^n, such that for

$$\frac{\partial q_i^n}{\partial x} = \frac{q_i^n - q_{i-1}^n}{\Delta x} + O(\Delta x)$$

We have the following TSE for q_{i-1}^n:

$$q_{i-1}^n = q(x_i - \Delta x, t^n) = q_i^n - \frac{\partial q_i^n}{\partial x}\Delta x + \frac{\partial^2 q_i^n}{\partial x^2}\frac{(\Delta x)^2}{2!} - O(\Delta x^3)$$

Substituting q_{i-1}^n back into Equation 2.1, we then get

$$\frac{\partial q_i^n}{\partial x} = \frac{q_i^n - \left(q_i^n - \frac{\partial q_i^n}{\partial x}\Delta x + \frac{\partial^2 q_i^n}{\partial x^2}\frac{(\Delta x)^2}{2!} - O(\Delta x^3) \right)}{\Delta x}$$

$$= \frac{\frac{\partial q_i^n}{\partial x}\Delta x - \frac{\partial^2 q_i^n}{\partial x^2}\frac{(\Delta x)^2}{2!} + O(\Delta x^3)}{\Delta x} = \frac{\partial q_i^n}{\partial x} - \underbrace{\frac{\partial^2 q_i^n}{\partial x^2}\frac{(\Delta x)}{2!} + O(\Delta x^2)}_{O(\Delta x)}$$

$$= \frac{\partial q_i^n}{\partial x} + O(\Delta x)$$

Therefore, we have shown that the backward difference formula for the first order partial derivative $q_x(x_i, t^n)$ is equal to the exact solution plus an error term on the order of $O(\Delta x)$.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B: 4TH ORDER, CENTERED DIFFERENCE STENCIL

We begin the construction of this FD stencil by first deciding which grid points we want to use, and then expanding about the point (x_i, t^n) using TSE.

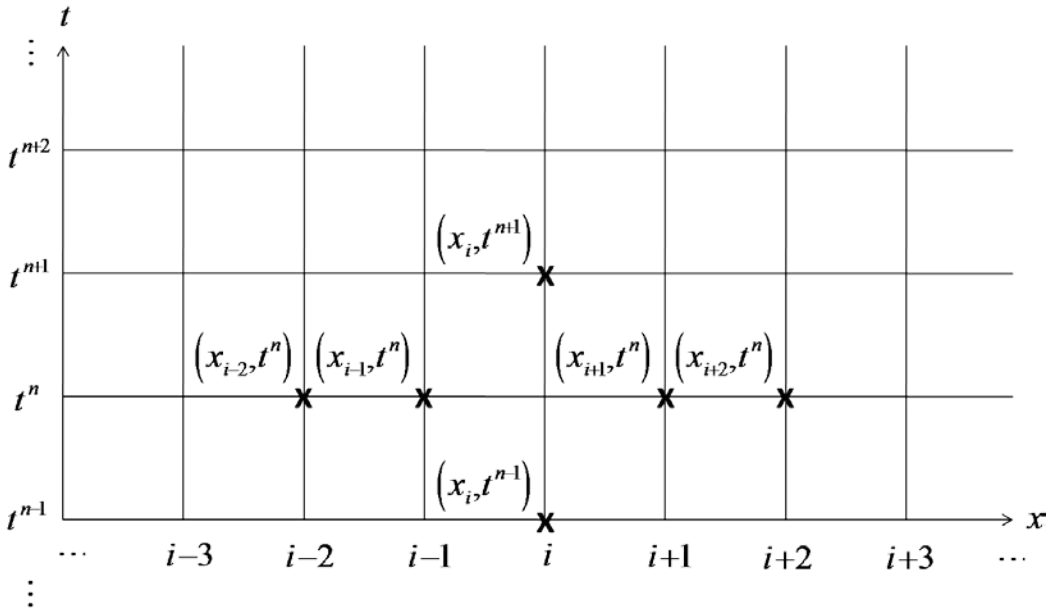

Figure 35. Uniform Grid in Space and Time

From Figure 35, we can see that there are many grid points to choose from when building a new FD method. However, for the construction of this 4th order, centered difference stencil, we will use the following grid points:

$$q\left(x_{i+1}, t^n\right) \quad , \quad q\left(x_{i+2}, t^n\right) \quad , \quad q\left(x_{i-1}, t^n\right) \quad , \quad q\left(x_{i-2}, t^n\right)$$

The TSE for each point are

$$q_{i+1}^n = q(x_i + \Delta x, t^n) = q_i^n + \frac{\partial q_i^n}{\partial x}\Delta x + \frac{\partial^2 q_i^n}{\partial x^2}\frac{(\Delta x)^2}{2!} + \frac{\partial^3 q_i^n}{\partial x^3}\frac{(\Delta x)^3}{3!} + O(\Delta x^4) \qquad (B.1)$$

$$q_{i+2}^n = q(x_i + 2\Delta x, t^n) = q_i^n + \frac{\partial q_i^n}{\partial x}2\Delta x + \frac{\partial^2 q_i^n}{\partial x^2}\frac{(2\Delta x)^2}{2!} + \frac{\partial^3 q_i^n}{\partial x^3}\frac{(2\Delta x)^3}{3!} + O[(2\Delta x)^4] \qquad (B.2)$$

$$q_{i-1}^n = q(x_i - \Delta x, t^n) = q_i^n - \frac{\partial q_i^n}{\partial x}\Delta x + \frac{\partial^2 q_i^n}{\partial x^2}\frac{(\Delta x)^2}{2!} - \frac{\partial^3 q_i^n}{\partial x^3}\frac{(\Delta x)^3}{3!} + O(\Delta x^4) \qquad (B.3)$$

$$q_{i-2}^n = q(x_i - 2\Delta x, t^n) = q_i^n - \frac{\partial q_i^n}{\partial x}2\Delta x + \frac{\partial^2 q_i^n}{\partial x^2}\frac{(2\Delta x)^2}{2!} - \frac{\partial^3 q_i^n}{\partial x^3}\frac{(2\Delta x)^3}{3!} + O[(2\Delta x)^4] \qquad (B.4)$$

Once we have expanded each point, we then need to consider the following sum of equations

$$\begin{aligned}
&\mathrm{A}\left[q_{i+1}^n = q_i^n + \frac{\partial q_i^n}{\partial x}\Delta x + \frac{\partial^2 q_i^n}{\partial x^2}\frac{(\Delta x)^2}{2!} + \frac{\partial^3 q_i^n}{\partial x^3}\frac{(\Delta x)^3}{3!} + O(\Delta x^4) \right] \\
&+ \mathrm{B}\left[q_{i+2}^n = q_i^n + \frac{\partial q_i^n}{\partial x}2\Delta x + \frac{\partial^2 q_i^n}{\partial x^2}\frac{(2\Delta x)^2}{2!} + \frac{\partial^3 q_i^n}{\partial x^3}\frac{(2\Delta x)^3}{3!} + O[(2\Delta x)^4] \right] \\
&+ \mathrm{C}\left[q_{i-1}^n = q_i^n - \frac{\partial q_i^n}{\partial x}\Delta x + \frac{\partial^2 q_i^n}{\partial x^2}\frac{(\Delta x)^2}{2!} - \frac{\partial^3 q_i^n}{\partial x^3}\frac{(\Delta x)^3}{3!} + O(\Delta x^4) \right] \\
&+ \mathrm{D}\left[q_{i-2}^n = q_i^n - \frac{\partial q_i^n}{\partial x}2\Delta x + \frac{\partial^2 q_i^n}{\partial x^2}\frac{(2\Delta x)^2}{2!} - \frac{\partial^3 q_i^n}{\partial x^3}\frac{(2\Delta x)^3}{3!} + O[(2\Delta x)^4] \right]
\end{aligned}$$

such that we must sum the columns of the system to determine what the coefficients *A*, *B*, *C*, and *D* must be. We do this by setting all columns containing the derivatives we do not want equal to zero. For example, we are building a FD stencil for the first derivative; therefore, the column containing the first partial derivative will be set equal to one and all others set equal to zero.

Summing the columns from the equations above gives us:

$$Aq_{i+1}^n + Bq_{i+2}^n + \mathrm{C}q_{i-1}^n + Dq_{i-2}^n \qquad (B.5)$$

$$q_i^n(A + B + \mathrm{C} + D) \qquad (B.6)$$

$$\frac{\partial q_i^n}{\partial x}\Delta x(A + 2B - C - 2D) \qquad (B.7)$$

$$\frac{\partial^2 q_i^n}{\partial x^2}\frac{(\Delta x)^2}{2!}(A + 4B + C + 4D) \qquad (B.8)$$

$$\frac{\partial^3 q_i^n}{\partial x^3}\frac{(\Delta x)^3}{3!}(A+8B-C-8D) \qquad (B.9)$$

$$O(\Delta x^4)(A+16B+C+16D) \qquad (B.10)$$

Now, in order to eliminate terms, we set equations (*B.8*), (*B.9*), and (*B.10*) equal to zero and Equation (*B.7*) equal to one, and then solve for *A, B, C,* and *D*, such that:

$$\left.\begin{array}{l}(A+2B-C-2D)=1\\(A+4B+C+4D)=0\\(A+8B-C-8D)=0\\(A+16B+C+16D)=0\end{array}\right\}\begin{array}{ll}12B+12D=0 & \rightarrow \quad B=-D\\-3A-3C=0 & \rightarrow \quad A=-C\end{array}$$

We can then substitute these two equations back into the first four to get

$$\left.\begin{array}{l}(2A-4D)=1\\(2A-16D)=0\end{array}\right\} \rightarrow \left.\begin{array}{l}A-2D=\frac{1}{2}\\A-8D=0\end{array}\right\} \rightarrow D=\frac{1}{12}, \quad A=\frac{2}{3}$$

Using these two values, we then get $C=-\frac{2}{3}$, and $B=-\frac{1}{12}$.

Another possible method for quickly solving the above system of equations is to use the method of LU decomposition by writing the system in matrix form $Ax=b$, and then solving for x, such that:

$$\begin{bmatrix}1 & 2 & -1 & -2\\1 & 4 & 1 & 4\\1 & 8 & -1 & -8\\1 & 16 & 1 & 16\end{bmatrix}.\begin{bmatrix}A\\B\\C\\D\end{bmatrix}=\begin{bmatrix}1\\0\\0\\0\end{bmatrix}, \qquad x=\begin{bmatrix}A\\B\\C\\D\end{bmatrix}.$$

We have now found the four coefficient values for our 4th order, centered difference stencil; therefore, we can now use Equations (*B.5*) - (*B.7*) to construct the FD stencil.

$$\frac{\partial q_i^n}{\partial x} = \frac{(B.5) - (B.6)}{\Delta x}$$

$$\frac{\partial q_i^n}{\partial x} = \frac{(Aq_{i+1}^n + Bq_{i+2}^n + \mathrm{C}q_{i-1}^n + Dq_{i-2}^n) - (q_i^n(A + B + \mathrm{C} + D))}{\Delta x}$$

$$\frac{\partial q_i^n}{\partial x} = \frac{\left(\frac{2}{3}q_{i+1}^n - \frac{1}{12}q_{i+2}^n - \frac{2}{3}q_{i-1}^n + \frac{1}{12}q_{i-2}^n\right) - (q_i^n(0))}{\Delta x}$$

Finally, we have our 4th order, centered difference stencil for the first derivative as

$$\frac{\partial q_i^n}{\partial x} = \frac{(q_{i-2}^n - 8q_{i-1}^n + 8q_{i+1}^n - q_{i+2}^n)}{12\Delta x} + O(\Delta x^4).$$

APPENDIX C: 2ND ORDER RK$_2$ PROOF

The following is a proof, which shows that regardless of the spatial discretization method, the order of accuracy for any time-integration method is independent of the spatial scheme. First let us begin with Equation (3.7),

$$k_1 = \Delta t F(t^n, q^n) \tag{C.0.1}$$

$$k_2 = \Delta t F(t^{n+1}, q^n + k_1), \tag{C.0.2}$$

$$q^{n+1} = q^n + \frac{1}{2}\left(k_1 + k_2\right) \tag{C.0.3}$$

and the following expression,

$$q_t^n = F(q^n), \tag{C.1}$$

where we can substitute C.0.1 and C.0.2 into C.0.3 to arrive at:

$$q^{n+1} = q^n + \frac{\Delta t}{2}\left(F(q^n) + F(q^n + \Delta t F(q^n))\right).$$

If we now use Taylor series expansion (TSE) for both the left and right hand sides of the above expression for RK$_2$ we achieve the following:

$$q^n + q_t^n \Delta t + q_{tt}^n \frac{\Delta t^2}{2} + O(\Delta t^3) = q^n + \frac{\Delta t}{2}\left(F(q^n) + F(q^n) + \frac{\partial F^n}{\partial q}\Delta t F(q^n) + O(\Delta t^2)\right)$$

$$q_t^n \Delta t + q_{tt}^n \frac{\Delta t^2}{2} + O(\Delta t^3) = F(q^n)\Delta t + \frac{\Delta t^2}{2}\frac{\partial F^n}{\partial q} F(q^n) + O(\Delta t^3)\ , \tag{C.2}$$

where, if we recognize that $\frac{\partial F^n}{\partial q} F(q^n)$ is the total derivative of $F = F(q(t))$ with respect to *t*, then

$$\frac{\partial F^n}{\partial q} F(q^n) = \frac{dF^n}{dt}, \quad \text{and} \quad \frac{dF^n}{dt} = q_{tt}^n .$$

Therefore, using this knowledge, we can simplify Equation (C.2) above to be

$$q_t^n = F(q^n) + O(\Delta t^2), \tag{C.3}$$

where we easily recognize that Equation (C.3) is equivalent to (C.1) with the addition of higher order terms. However, it must be shown that the terms on the order of $O(\Delta t^3)$ from (C.2) do not cancel each other out when we simplified the expression to Equation (C.3).

Therefore, if we look at the higher order terms on the LHS, we find that

$$O(\Delta t^3) = q_{ttt}^n \frac{\Delta t^3}{6} + O(\Delta t^4),$$

whereas the higher order terms on the RHS are equivalent to

$$O(\Delta t^3) = \frac{\Delta t^3}{4} \frac{\partial^2 F^n}{\partial q^2} F^2(q^n) + O(\Delta t^4).$$

If we assume these two expressions are equal to each other we find,

$$q_{ttt}^n \frac{\Delta t^3}{6} + O(\Delta t^4) = \frac{\Delta t^3}{4} \frac{\partial^2 F^n}{\partial q^2} F^2(q^n) + O(\Delta t^4). \tag{C.4}$$

Now, if we use the following equalities:

$$\frac{\partial q^n}{\partial t} = F(q^n), \qquad \frac{\partial^2 q^n}{\partial t^2} = \frac{\partial F^n}{\partial q} F(q^n), \qquad \frac{\partial^3 q^n}{\partial t^3} = \frac{\partial^2 F^n}{\partial q^2} F^2(q^n) + F(q^n)\left(\frac{\partial F^n}{\partial q}\right)^2,$$

then substituting into Equation (C.4),

$$\left(\frac{\partial^2 F^n}{\partial q^2} F^2(q^n) + F(q^n)\left(\frac{\partial F^n}{\partial q}\right)^2\right)\frac{\Delta t^3}{6} + O(\Delta t^4) \neq \frac{\Delta t^3}{4} \frac{\partial^2 F^n}{\partial q^2} F^2(q^n) + O(\Delta t^4),$$

we find that these two expressions are not equal to each other; therefore, showing that the higher order terms in Equation (C.2) do not cancel each other, and proving that by using RK_2, Equation (C.3) is equivalent to (C.1) with second order accuracy.

APPENDIX D: DERIVATION OF AB_2 AND BDF_2

In order to derive AB_2, we begin with

$$\frac{\partial q}{\partial t} = F(q),$$

and assume we want to find

$$\frac{\partial q^{n+\frac{1}{2}}}{\partial t} = F\left(q^{n+\frac{1}{2}}\right).$$

Using TSE to expand about $t^{n+\frac{1}{2}}$ for the solutions at q^{n+1} and q^n yields:

$$q^{n+1} = q^{n+\frac{1}{2}} + \frac{\Delta t}{2} q_t^{n+\frac{1}{2}} + \frac{\left(\Delta t/2\right)^2}{2} q_{tt}^{n+\frac{1}{2}} + O(\Delta t^3)$$
$$q^{n} = q^{n+\frac{1}{2}} - \frac{\Delta t}{2} q_t^{n+\frac{1}{2}} + \frac{\left(\Delta t/2\right)^2}{2} q_{tt}^{n+\frac{1}{2}} + O(\Delta t^3).$$

Next, isolating $q_t^{n+\frac{1}{2}}$ provides us with,

$$\Delta t q_t^{n+\frac{1}{2}} = q^{n+1} - q^n + O(\Delta t^3)$$

$$F\left(q^{n+\frac{1}{2}}\right) = \frac{\partial q^{n+\frac{1}{2}}}{\partial t} = \frac{q^{n+1} - q^n}{\Delta t} + O(\Delta t^2)\ .$$

Next, let us expand $F\left(q^n\right)$ and $F\left(q^{n-1}\right)$ about $q^{n+\frac{1}{2}}$ to arrive at:

$$F\left(q^n\right) = F\left(q^{n+\frac{1}{2}} - \frac{\Delta t}{2} q_t^{n+\frac{1}{2}} + O(\Delta t^2)\right)$$
$$F\left(q^{n-1}\right) = F\left(q^{n+\frac{1}{2}} - \frac{3\Delta t}{2} q_t^{n+\frac{1}{2}} + O(\Delta t^2)\right).$$

Now, taking a TSE about $F\left(q^{n+\frac{1}{2}}\right)$, we find that,

$$F\left(q^{n}\right)=F\left(q^{n+\frac{1}{2}}\right)-\frac{\partial F^{n+\frac{1}{2}}}{\partial q}\frac{\Delta t}{2}q_{t}^{n+\frac{1}{2}}+O(\Delta t^{2})$$

$$F\left(q^{n-1}\right)=F\left(q^{n+\frac{1}{2}}\right)-\frac{\partial F^{n+\frac{1}{2}}}{\partial q}\frac{3\Delta t}{2}q_{t}^{n+\frac{1}{2}}+O(\Delta t^{2}),$$

where in order to eliminate $O(\Delta t)$ terms, requires:

$$3F\left(q^{n}\right)-F\left(q^{n-1}\right)=2F\left(q^{n+\frac{1}{2}}\right)+O(\Delta t^{2}).$$

Therefore, we can finally show that,

$$F\left(q^{n+\frac{1}{2}}\right)=\frac{3}{2}F\left(q^{n}\right)-\frac{1}{2}F\left(q^{n-1}\right)+O(\Delta t^{2})$$

$$\frac{q^{n+1}-q^{n}}{\Delta t}+O(\Delta t^{2})=\frac{3}{2}F\left(q^{n}\right)-\frac{1}{2}F\left(q^{n-1}\right)+O(\Delta t^{2})$$

$$\frac{q^{n+1}-q^{n}}{\Delta t}=\frac{3}{2}F\left(q^{n}\right)-\frac{1}{2}F\left(q^{n-1}\right)+O(\Delta t^{2})$$

$$q^{n+1}=q^{n}+\frac{\Delta t}{2}\left(3F\left(q^{n}\right)-F\left(q^{n-1}\right)\right)+O(\Delta t^{3}).$$

This concludes the construction of AB_2 using TSE and matches Equation (3.2).

In contrast to the AB_P methods, which expand about $q^{n+\frac{1}{2}}$, the BDF_P methods are fully implicit with the RHS at q^{n+1}. First, let us take

$$\frac{\partial q^{n+1}}{\partial t}=F\left(q^{n+1}\right),$$

where if we use TSE to expand q^{n} and q^{n-1} about q^{n+1}, we have the following:

$$q^{n}=q^{n+1}-\Delta t q_{t}^{n+1}+\frac{\Delta t^{2}}{2}q_{tt}^{n+1}+O(\Delta t^{3})$$

$$q^{n-1}=q^{n+1}-2\Delta t q_{t}^{n+1}+\frac{\left(2\Delta t\right)^{2}}{2}q_{tt}^{n+1}+O(\Delta t^{3}).$$

Next, eliminating the terms $O(\Delta t^2)$ requires $4q^n - q^{n-1}$:

$$4q^n - q^{n-1} = 4\left[q^{n+1} - \Delta t q_t^{n+1} + \frac{\Delta t^2}{2} q_{tt}^{n+1} + O(\Delta t^3) \right] - \left[q^{n+1} - 2\Delta t q_t^{n+1} + \frac{(2\Delta t)^2}{2} q_{tt}^{n+1} + O(\Delta t^3) \right]$$

$$4q^n - q^{n-1} = 3q^{n+1} - 2\Delta t q_t^{n+1} + O(\Delta t^3).$$

If we now solve for q_t^{n+1}, then

$$4q^n - q^{n-1} = 3q^{n+1} - 2\Delta t q_t^{n+1} + O(\Delta t^3)$$

$$q_t^{n+1} = \frac{\frac{3}{2}q^{n+1} - 2q^n + \frac{1}{2}q^{n-1}}{\Delta t} + O(\Delta t^2).$$

Now, if we substitute the TSE for q^n and q^{n-1} into $F\left(q^n\right)$ and $F\left(q^{n-1}\right)$, then we have

$$F\left(q^n\right) = F\left(q^{n+1} - \Delta t q_t^{n+1} + \frac{\Delta t^2}{2} q_{tt}^{n+1} + O(\Delta t^3) \right)$$

$$F\left(q^{n-1}\right) = F\left(q^{n+1} - 2\Delta t q_t^{n+1} + \frac{(2\Delta t)^2}{2} q_{tt}^{n+1} + O(\Delta t^3) \right)$$

Again, using TSE to expand $F\left(q^n\right)$ and $F\left(q^{n-1}\right)$ about $F\left(q^{n+1}\right)$, we then achieve

$$F\left(q^n\right) = F\left(q^{n+1}\right) + \frac{\partial F}{\partial q}^{n+1} \Delta t q_t^{n+1} + O(\Delta t^2)$$

$$F\left(q^{n-1}\right) = F\left(q^{n+1}\right) + \frac{\partial F}{\partial q}^{n+1} 2\Delta t q_t^{n+1} + O(\Delta t^2),$$

where we want to eliminate terms of $O(\Delta t)$, which requires $2F\left(q^n\right) - F\left(q^{n-1}\right)$, such that

$$F\left(q^{n+1}\right) = 2F\left(q^n\right) - F\left(q^{n-1}\right) + O(\Delta t^2).$$

Now, since we know that $F\left(q^{n+1}\right) = q_t^{n+1}$, we can rewrite the above as follows:

$$F\left(q^{n+1}\right) = q_t^{n+1} = \frac{\frac{3}{2}q^{n+1} - 2q^n + \frac{1}{2}q^{n-1}}{\Delta t} + O(\Delta t^2) = 2F\left(q^n\right) - F\left(q^{n-1}\right) + O(\Delta t^2).$$

After simplifying the above equation, we finally have the expression for BDF_2:

$$q^{n+1} = \frac{4}{3}q^n - \frac{1}{3}q^{n-1} + \frac{2\Delta t}{3}\left(2F(t^n, q^n) - F(t^{n-1}, q^{n-1})\right),$$

which is equivalent to Equation (3.6).

It is also important to note that neither derivation for AB_2 or BDF_2 required any knowledge of how we intend to solve for the RHS $F\left(q^n\right)$, or $F\left(q^{n-1}\right)$. In fact, we have already seen in Chapter II, that we can represent the right hand side by a wide variety of difference schemes, depending on the order of accuracy we want to achieve for the spatial derivative.

LIST OF REFERENCES

[1] F. Giraldo, and M. Restelli. “A study of spectral element and discontinuous galerkin methods for the navier-stokes equations in non-hydrostatic mesoscale atmosphere modeling: equation sets and test cases,” *Journal of Computational Physics*, vol. 227, pp. 3849–3877, 2008.

[2] M. Restelli, and F. Giraldo, “A conservative discontinuous galerkin semi-implicit formulation for the navier-stokes equations in non-hydrostatic mesoscale modeling.” *SIAM Journal of Scientific Computing*, vol. 31, pp. 2231–2257, 2009.

[3] D. A. Anderson, J. C. Tannehill and R. H. Pletcher, *Computational Fluid Mechanics and Heat Transfer.* New York, N.Y.: Hemisphere Pub. Corp., 1984.

[4] M. L. Ghrist, “High-order finite difference methods for wave equations,” Ph.D. dissertation, University of Colorado, Denver, CO, 2000.

[5] Z. Li, L. Vulkov and J. Waśniewski, *Numerical Analysis and its Applications: Third International Conference, NAA 2004, Rousse, Bulgaria, June 29-July 3, 2004: Revised Selected Papers.* Berlin; New York: Springer, 2005.

[6] E. T. Whittaker and G. N. Watson, *A Course of Modern Analysis: An Introduction to the General Theory of Infinite Processes and of Analytic Functions; with an Account of the Principal Transcendental Functions.* Cambridge University Press, 1927.

[7] P. D. Lax, *Hyperbolic Partial Differential Equations.* Providence, R.I.: American Mathematical Society, 2006.

[8] “Element based galerkin methods,” class notes for MA 4245, F. X. Giraldo, Department of Applied Mathematics, Naval Postgraduate School, Spring 2012.

[9] D. R. Durran, *Numerical Methods for Wave Equations in Geophysical Fluid Dynamics.* New York: Springer, 1999.

[10] J. C. Butcher, *Numerical Methods for Ordinary Differential Equations*. West Sussex, England: John Wiley & Sons, 2008.

[11] E. Hairer, S. P. Nørsett and G. Wanner, *Solving Ordinary Differential Equations.* Berlin; New York: Springer-Verlag, 1987.

[12] R. L. Burden, and J. D. Faires, *Numerical Analysis.* Boston, Mass.: Brooks/Cole, 2005.

[13] E. M. Constantinescu and A. Sandu, *Multirate Timestepping Methods for Hyperbolic Conservation Laws*. Springer Science and Business Media, LLC, 2007.

[14] C. W. Shu, and S. Osher, "Efficient implementation of essentially non-oscillatory shock-capturing schemes," *Journal of Computational Physics*, vol. 77, pp. 439-471. 1988.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. Prof. Carlos Borges
   Naval Postgraduate School
   Monterey, California

4. Prof. Francis Giraldo
   Naval Postgraduate School
   Monterey, California

5. Assoc. Prof. Hong Zhou
   Naval Postgraduate School
   Monterey, California